AREA ATTENTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing attention mechanisms, are mostly point-based in that a model is designed to attend to a single item in a collection of items (the memory). Intuitively, an area in the memory that may contain multiple items can be worth attending to as well. Although Softmax, which is typically used for computing attention alignments, assigns non-zero probability for every item in memory, it tends to converge to a single item and cannot efficiently attend to a group of items that matter. We propose area attention: a way to attend to an area of the memory, where each area contains a group of items that are either spatially adjacent when the memory has a 2-dimensional structure, such as images, or temporally adjacent for 1-dimensional memory, such as natural language sentences. Importantly, the size of an area, i.e., the number of items in an area, can vary depending on the learned coherence of the adjacent items. Using an area of items, instead of a single, we hope attention mechanisms can better capture the nature of the task. Area attention can work along multi-head attention for attending multiple areas in the memory. We evaluate area attention on two tasks: character-level neural machine translation and image captioning, and improve upon strong (state-of-the-art) baselines in both cases. In addition to proposing the novel concept of area attention, we contribute an efficient way for computing it by leveraging the technique of summed area tables.

1 INTRODUCTION

Attentional mechanisms have significantly boosted the accuracy on a variety of deep learning tasks (Bahdanau et al., 2014; Luong et al., 2015; Xu et al., 2015). They allow the model to selectively focus on specific pieces of information, which can be a word in a sentence for neural machine translation (Bahdanau et al., 2014; Luong et al., 2015) or a region of pixels in image captioning (Xu et al., 2015; Soricut et al., 2018).

An attentional mechanism typically follows a *memory-query* paradigm, where the memory M contains a collection of items of information from a source modality such as the embeddings of an image or the hidden states of encoding an input sentence, and the query q comes from a target modality such as the hidden state of a decoder model. In recent architectures such as Transformer (Vaswani et al., 2017), self-attention involves queries and memory from the same modality for either encoder or decoder. Each item in the memory has a key and value (k_i, v_i), where the key is used to compute the probability a_i regarding how well the query matches the item (see Equation 1).

$$a_i = \frac{\exp(f_{att}(q, k_i))}{\sum_{j=1}^{|M|} \exp(f_{att}(q, k_j))} \tag{1}$$

The typical choices for f_{att} include dot products qk_i (Luong et al., 2015) and a multilayer perceptron (Bahdanau et al., 2014). The output O_q^M from querying the memory M with q is then calculated as the sum of all the values in the memory weighted by their probabilities (see Equation 2), which can be fed to other parts of the model for further calculation. During training, the model learns to attend to specific piece of information, e.g., the correspondance between a word in the target sentence and a word in the source sentence for translation tasks.

$$O_q^M = \sum_{i=1}^{|M|} a_i v_i \tag{2}$$

Attention mechanisms are designed to focus on a single item in the entire memory. Although softmax (Equation 1) assigns non-zero probablities to every item in the memory, it quickly converges to the most probable item due to the exponential function used for calculating probabilities. While this kind of single-item attention has been helpful for many tasks, it is fundamentally limited for modeling complex attention distribution that might be involved in a task.

To combat this issue, multi-head attention (Vaswani et al., 2017) allows a model to attend multiple items at the same time where each head captures a different aspect of the task, although each head is still single-item-based attention. In this paper, we propose *area attention*, which given a query, allows the model to attend to a group of items in the memory that are structurally adjacent. In area attention, each unit for attention calculation is an area that can have one or more than one item. Each of these areas can have a varying number of items that allow the model to capture rich alignment distributions. Area attention is complementary to multi-head attention. Each head can be using area attention and multi-head area attention allows the model to attend to multiple areas. As we show in the experiments, the combination of both achieved the best results.

We evaluated area attention on two tasks: character-level machine translation on both English to German and English to French (Wu et al., 2016; Lee et al., 2016), and image captioning (Soricut et al., 2018). These tasks involve multiple model architectures, including the canonical LSTM seq2seq with attention (Luong et al., 2015) and the encoder-decoder Transformer (Vaswani et al., 2017).

2 Area-Based Attention Mechanisms

An area is a group of structurally adjacent items in the memory. When the memory consists of a sequence of items, a 1-dimensional structure, an area is a range of items that are sequentially (or temporally) adjacent and the number of items in the area can be one or multiple. Many language-related tasks are categorized in the 1-dimensional case, e.g., machine translation or sequence prediction tasks. In Figure 1, the original memory is a 4-item sequence. By combining the adjacent items in the sequence, we form area memory where each item is a combination of multiple adjacent items in the original memory. We can limit the maximum area size to consider for a task. In Figure 1, the maximum area size is 3.

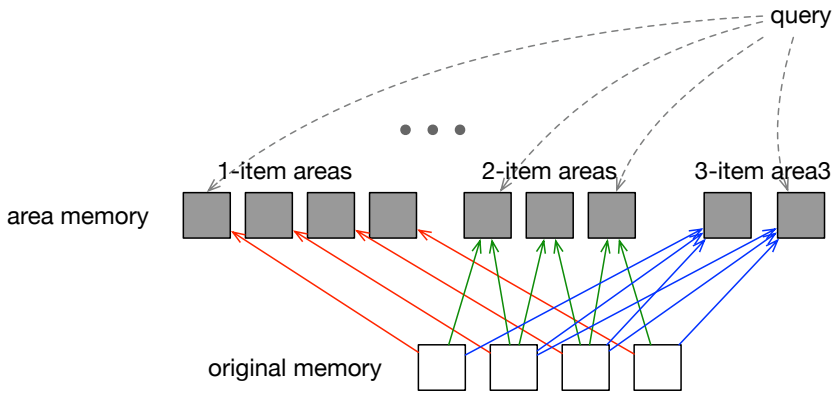

Figure 1: An illustration of area attention for the 1-dimensional case. In this example, the memory is a 4-item sequence and the maximum size of an area allowed is 3.

When the memory contains a grid of items, a 2-dimensional structure, an area can be any rectangular region in the grid (see Figure 2). This resembles many image-related tasks, e.g., image captioning. Again, we can limit the maximum size allowed for an area. For a 2-dimensional area, we can set the maximum height and width for each area. In this example, the original memory is a 3x3 grid of items and the maximum height and width allowed for each area is 2.

As we can see, many areas can be generated by combining adjacent items. For the 1-dimensional case, the number of areas that can be generated is $|R| = (L - S)S + (S + 1)S/2$ where S is the maximum size of an area and L is the length of the sequence. For the 2-dimensional case, there are an quadratic number of areas can be generated from the original memory: $|R| = |R_v||R_h|$.

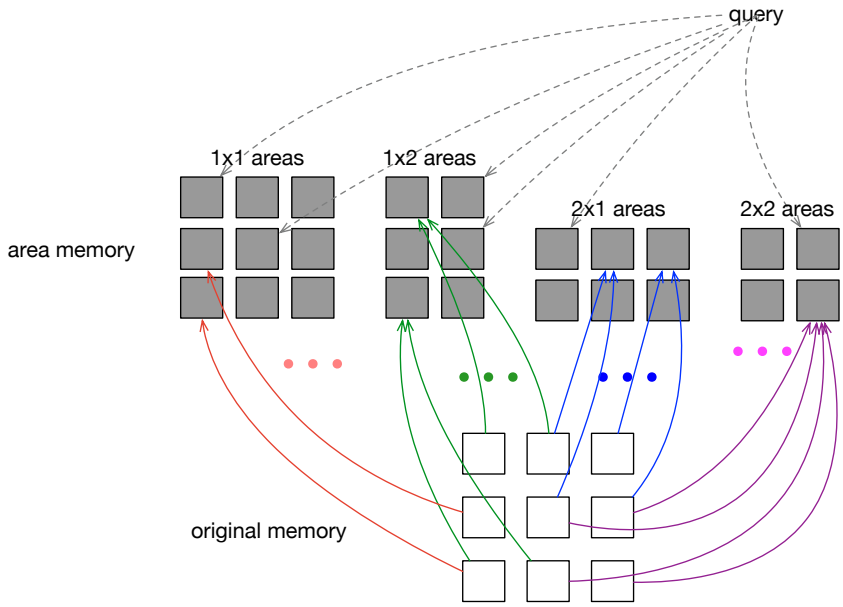

Figure 2: An illustration of area attention for the 2-dimensional case. In this example, the memory is a 3x3 grid and the dimension allowed for an area is 2x2.

$|R_v| = (L_v - H)H + (H+1)H/2$ and $|R_h| = (L_h - W)W + (W+1)W/2$ where L_v and L_h are the height and width of the memory grid and H and W are the maximum height and width allowed for a rectangular area.

To be able to attend to each area, we need to define the key and value for each area that contains multiple items in the original memory. As the first step to explore area attention, we define the key of an area, μ_i, simply as the mean vector of the key of each item in the area.

$$\mu_i = \frac{1}{|r_i|} \sum_{j=1}^{|r_i|} k_{i,j} \tag{3}$$

where $|r_i|$ is the size of the area r_i. For the value of an area, we simply define it as the the sum of all the value vectors in the area.

$$v_i^{r_i} = \sum_{j=1}^{|r_i|} v_{i,j} \tag{4}$$

With the keys and values defined, we can use the standard way for calculating attention as discussed in Equation 1 and Equation 2.

2.1 Combining Area Features

Alternatively, we can derive a richer representation of each area by using features other than the mean of the key vectors of the area. For example, we can consider the variance of the key vectors within each area.

$$\sigma_i = \frac{1}{|r_i|} \sum_{l=1}^{|r_i|} (k_{i,l} - \mu_i)^2 \tag{5}$$

We can also consider the size of each area s_i,$1 \leq s_i \leq |R|$, as a feature of the area. To combine these features, a simple way is to use the concatenation of these features. In our work, we project s_i to a vector space that has the same depth as μ_i and σ_i, using size embedding (Equation 6).

$$e_i = 1(s_i)E^s \tag{6}$$

where $1(s_i)$ is a one-hot encoding of s_i and $E^s \in R^{|R| \times D}$ is the embedding matrix, where D is the depth of the key vector. With the mean, variance and size having the same depth, we can compute the key of an area k_i by combining these features in the following way.

$$k_i^r = W_d \phi(\mu_i + \sigma_i + e_i; \theta) \tag{7}$$

where ϕ is a one-layer perceptron using a nonlinear transformation such as ReLU, and $W_d \in \mathbb{R}^{D \times D}$. Both θ and W_d are trainable parameters.

2.2 Fast Computation Using Summed Area Table

If we naively compute μ_i, σ_i and $v_i^{r_i}$, the time complexity for computing attention will be $O(|M|A^2)$ where $|M|$ is the size of the memory that is L for a 1-dimensional sequence or $L_v L_h$ for a 2-dimensional memory. A is the maximum size of an area, which is S in the one dimensional case and WH in the 2-dimensional case. This is computationally expensive in comparison to the attention computed on the original memory, which is $O(|M|)$. To address the issue, we use summed area table, an optimization technique that has been used in computer vision for computing features on image areas (Viola & Jones, 2001). It allows constant time to calculate a summation-based feature in each rectangular area, which allows us to bring down the time complexity to $O(|M|A)$.

Summed area table is based on a pre-computed integral image, I, which can be computed in a single pass of the memory. Here let us focus on the area value calculation for a 2-dimensional memory because a 1-dimensional memory is just a special case with the height of the grid as 1.

$$I_{x,y} = v_{x,y} + I_{x,y-1} + I_{x-1,y} \tag{8}$$

where x and y are the coordinates of the item in the memory. With the integral image, we can calculate the key and value of each area in constant time. The sum of all the vectors in a rectangular area can be easily computed as the following.

$$v_{x1,y1,x2,y2} = I_{x2,y2} + I_{x1,y1} - I_{x2,y1} - I_{x1,y2} \tag{9}$$

where $v_{x1,y1,x2,y2}$ is the value for the area located with the top-left corner at (x_1, y_1) and the bottom-right corner at (x_2, y_2). In a similar way, we can compute $\mu_{x1,y1,x2,y2}$. $\sigma_{x1,y1,x2,y2}$ can also be computed at constant time for each area (Equation 10).

$$\sigma_{x1,y1,x2,y2} = \frac{(I_{x2,y2})^2 + (I_{x1,y1})^2 - (I_{x2,y1})^2 - (I_{x1,y2})^2}{(x2 - x1) \times (y2 - y1)} - (\mu_{x1,y1,x2,y2})^2 \tag{10}$$

The core component for computing these quantities is to be able to quickly compute the sum of vectors in each area after we obtain the integral image table I for each coordinate $[x, y]$ as shown in Eq(8) using cumulative sum. We present the Pseudo code for performing Equation 8 and Equation 9 in Algorithm (1) and the code for computing the mean, sum and variance (Equation 10) of each area in Algorithm (2). These Pseudo code are designed based on Tensor operations, which can be implemented efficiently using libraries such as TensorFlow [1] and PyTorch [2].

[1] https://github.com/tensorflow/tensorflow

[2] https://github.com/pytorch/pytorch

Algorithm 1: Compute the vector sum and the size of each area, for all the qualified rectangular areas on a given grid.

Input: A 3D tensor G that represents a grid where each item is a vector.
Output: Sum of vectors of each area, U, and corresponding size of each area, S.
Hyperparameter: maximum area width W and height H allowed.

Acquire G^P by padding all-zero vectors to the left and top of G. ;
Compute integral image I by using cumulative sum on both dimensions over G;
for $w = 1, \cdots, W$ **do**
  for $h = 1, \cdots, H$ **do**
    $I_1 \leftarrow I[h+1:, w+1:, :]$;
    $I_2 \leftarrow I[: -h-1, : -w-1, :]$;
    $I_3 \leftarrow I[h+1:, : -w-1, :]$;
    $I_4 \leftarrow I[: -h-1, w+1:, :]$;
    $\bar{U} = I_1 + I_2 - I_3 - I_4$;
    $\bar{S} \leftarrow [w \times h]$;
    $S \leftarrow [S \;\; \bar{S}]$;
    $U \leftarrow [U \;\; \bar{U}]$;
return U and S.

Algorithm 2: Compute the vector mean, variance, and sum as well as the size of each area, for all the qualified rectangular areas on a grid

Input: A grid of items G where each item is a vector.
Output: Vector mean μ, variance σ and sum U as well as size S of each area.

Acquire U and S using Algorithm 1 with input G.;
Acquire $U^{'}$ using Algorithm 1 with input $G \odot G$ that $\odot$ is for element-wise multiplication;
$\mu \leftarrow U \oslash S$ where $\oslash$ is for element-wise division;
$\mu^{'} \leftarrow U^{'} \oslash S$;
$\sigma \leftarrow \mu^{'} - \mu \odot \mu$;
return μ, σ, U, and S.

3 EXPERIMENTS

We experimented with area attention on two important tasks: character-level machine translation and image captioning, where attention has been an important component in these tasks. The architectures involves several popular encoder and decoder choices, such as LSTM (Hochreiter & Schmidhuber, 1997) and Transformer (Vaswani et al., 2017). The attention mechansims in these tasks include both self attention and encoder-decoder attention.

3.1 CHARACTER-LEVEL MACHINE TRANSLATION

For character-level machine translation tasks, we experimented with area attention on two model architectures. One is Transformer, a recent architecture (Vaswani et al., 2017) that has established the state of art for word-level machine translation tasks on WMT 2014 English-to-German and English-to-French tasks. The model has not been used for character-level translation tasks previously. The other architecture is LSTM with encoder-decoder attention, which has been a popular choice for machine translation tasks. We used the same datasets as the those used in (Vaswani et al., 2017).

In the WMT 2014 English-German dataset, there are about 4.5 million English-German sentence pairs, and in the English-French dataset, there are about 36 million English-French sentence pairs (Wu et al., 2016). During training, sentence pairs were batched together based on their approximate sequence lengths. Each training batch contained a set of sentence pairs that amount to approximately 32000 source characters and 32000 target characters.

3.1.1 TRANSFORMER EXPERIMENTS

Transformer heavily uses attentional mechanisms, including self-attention in both the encoder and the decoder, and the encoder-decoder attention. We vary the configuration of Transformer to see how area attention impacts the model. In particular, we experimented the following variations of Transformer (see Table 1). When using area attention, we applied area attention onto the first two layers of the encoder and decoder including both self-attention in them and the encoder-decoder attention. The maximum area size that we used in these experiments was 5.

We trained each of these models on one machine with 8 NVIDIA P100 GPUs for a total of 250,000 steps. All the training took less than 2 days to finish. Similar to previous work, we used the Adam optimizer with a varying learning rate over the course of training, as in (Vaswani et al., 2017) (see there for details).

Table 1: Transformer model variations in the experiments.

Configuration	#Hidden Layers	Hidden Size	Filter Size	#Attention Heads
Tiny	2	128	512	4
Small	2	256	1024	4
Base	6	512	2048	8

We found area attention consistently improved Transformer across all the model configurations. The benefit of area attention is particularly significant when the model is relatively small. For the character-level English-to-German translation, the best result we found in the literature is $BLEU = 22.62$ reported by (Wu et al., 2016). We achieved $BLEU = 25.03$ for the English-to-German character-level translation task and $BLEU = 33.69$ on the English-to-French character-level translation task. Note that these accuracy gains are based on the base form of area attention (see Equation 3 and 4), which does not add any additional training parameters to the model.

Table 2: The BLEU scores on test data for the Transformer-based architecture with a varying number of layers in both the encoder and decoder and a varying number of heads for multi-head self and encoder-decoder attention.

Model Configuration	Regular Attention		Area Attention (Eq.3 and 4)	
	EN-DE	EN-FR	EN-DE	EN-FR
Tiny	6.97	9.47	**7.39**	**11.79**
Small	12.18	18.75	**13.44**	**21.24**
Base	24.65	32.80	**25.03**	**33.69**

3.1.2 LSTM EXPERIMENTS

For LSTM, we used a 2-layer LSTM for both its encoder and decoder. The encoder-decoder attention is based on multiplicative attention where the alignment of a query and a memory key is computed as their dot product (Luong et al., 2015). We vary the size of LSTM to investigate how area attention can improve LSTM on the character-level translation tasks. Compared to Transformer, LSTM trains quite slow. We trained each of these models on one machine with 8 NVIDIA P100 GPUs over three days. For the purpose of observing the impact of area attention on each LSTM configuration, rather than for competing with Transformer, we trained LSTM for 80000 iterations.

LSTM enhanced with area attention outperformed the baseline in most conditions (see Table 3) on both the English-to-German and English-to-French character-level translation tasks. The accuracy gain on the English-to-French task seems more pronounced.

3.2 IMAGE CAPTIONING

Image captioning is the task to generate natural language description of an image that reflects the visual content of an image. This task has been addressed previously using a deep architecture that

Table 3: The Negative Log perplexity on predicting each character in a target sentence on validation data for the LSTM-based architecture with a varying number of LSTM cells. The larger the number is the better.

#LSTM Cells	Regular Attention		Area Attention (Eq.3 and 4)	
	EN-DE	EN-FR	EN-DE	EN-FR
256	-0.7984	-0.7770	**-0.7981**	**-0.7655**
512	-0.6506	-0.6343	**-0.6504**	**-0.6318**
1024	**-0.5585**	-0.5527	-0.5603	**-0.5521**

features an image encoder and a language decoder (Xu et al., 2015; Soricut et al., 2018). The image encoder typically employs a convolutional net such as ResNet (He et al., 2015) to embed the images and then uses a recurrent net such as LSTM or Transformer (Soricut et al., 2018) to encode the image based on these embeddings. For the decoder, either LSTM (Xu et al., 2015) or Transformer (Soricut et al., 2018) has been used for generating natural language descriptions. In many of these designs, attention mechanisms have been an important component that allows the decoder to selectively focus on a specific part of the image at each step of decoding, which often leads to better captioning quality.

In this experiment, we follow a champion condition in the experimental setup of (Soricut et al., 2018) that achieved state-of-the-art results. It uses a pre-trained Inception-ResNet to generate 8×8 image embeddings, a 6-layer Transformer for image encoding and a 6-layer Transformer for decoding. The dimension of Transformer is 512 and the number of heads is 8. We want to see how area attention improves the captioning accuracy, particularly regarding self-attention and encoder-decoder attention computed off the image, which resembles a 2-dimensional case for using area attention. We also vary the maximum area size allowed to see the impact.

Similar to (Soricut et al., 2018), we trained each model based on the training & development sets provided by the COCO dataset (Lin et al., 2014), which as 82K images for training and 40K for validation. Each of these images have at least 5 groudtruth captions. The training was conducted on a distributed learning infrastructure (Dean et al., 2012) with 10 GPU cores where updates are applied asynchronously across multiple replicas. We then tested each model on the Flickr 1K (Young et al., 2014) test set, which is out-of-domain for the trained model. For each experiment, we report CIDEr (Vedantam et al., 2014) and ROUGE-L (Lin & Och, 2004) metrics. For both metrics, higher number means better captioning accuracy—the closer distances between the predicted and the groundtruth captions.

In the benchmark model, a regular multi-head attention is used (see the first row in Table 4. We then experimented with several variations by adding area attention with different maximum area sizes to the first 2 layers of the image encoder and the first 2 layers of the caption decoder. 2×2 stands for the maximum area size for the first two layers in the image encoder (a 2-dimensional case). $3 \times 3^*$ denotes the maximum area size and $*$ means the combined area features are used for the key (Equation 7). 2^* stands for 1 dimensional area attention for the caption decoder.

Table 4: Test accuracy of image captioning models that are trained on COCO and tested on Flickr. See the previous results of the benchmark model at the row "T2T8x8 COCO" in Table 7 of (Soricut et al., 2018).

Self & Enc-Dec Attention on Image	Self-Attention on Caption	ROUGE-L	CIDEr
Regular	Regular	0.409	0.355
2×2 Eq. 3	Regular	0.410	0.359
$3 \times 3^*$ Eq. 7	Regular	0.419	**0.361**
$3 \times 3^*$ Eq. 7	2^* Eq. 7	**0.421**	0.360

We found models with area attention outperformed the benchmark on both ROUGE-L and CIDEr metrics. The model with 2×2 does not use any additional parameters–the same number of parameters as the benchmark. The model with $3 \times 3^*$ and 2^* achieved the best results, which only added a small

fraction of the number of parameters to the benchmark model. In comparison with the results reported in (Soricut et al., 2018) (Row "T2T8x8 COCO" in Table 7), our model advanced the benchmark accuracy on the task.

4 CONCLUSIONS

In this paper, we present a new way for calculating attention by attending to whole areas. An area contains a group of items in the memory to be attended. The items in the area are either spatially adjacent when the memory has 2-dimensional structure, such as images, or temporally adjacent for 1-dimensional memory, such as natural language sentences. Importantly, the size of an area, i.e., the number of items in an area, can vary depending on the learned coherence of the adjacent items. Area attention contrasts with the existing attentional mechanisms that are mostly point-based. We evaluated area attention on two tasks: character-level neural machine translation and image captioning, which involve model architectures such as Transformer and LSTM. Both on character-level translation and on image captioning, we obtained new state-of-the-art results using area attention.

REFERENCES

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014. URL http://arxiv.org/abs/1409.0473.

Jeffrey Dean, Greg S. Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Quoc V. Le, Mark Z. Mao, Marc'Aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, and Andrew Y. Ng. Large scale distributed deep networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems*, NIPS'12, pp. 1223–1231, USA, 2012. Curran Associates Inc. URL http://dl.acm.org/citation.cfm?id=2999134.2999271.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. URL http://arxiv.org/abs/1512.03385.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL http://dx.doi.org/10.1162/neco.1997.9.8.1735.

Jason Lee, Kyunghyun Cho, and Thomas Hofmann. Fully character-level neural machine translation without explicit segmentation. *CoRR*, abs/1610.03017, 2016. URL http://arxiv.org/abs/1610.03017.

Chin-Yew Lin and Franz Josef Och. Orange: A method for evaluating automatic evaluation metrics for machine translation. In *Proceedings of the 20th International Conference on Computational Linguistics*, COLING '04, Stroudsburg, PA, USA, 2004. Association for Computational Linguistics. doi: 10.3115/1220355.1220427. URL https://doi.org/10.3115/1220355.1220427.

Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. URL http://arxiv.org/abs/1405.0312.

Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. *CoRR*, abs/1508.04025, 2015. URL http://arxiv.org/abs/1508.04025.

Radu Soricut, Nan Ding, Piyush Sharma, and Sebastian Goodman. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pp. 2556–2565, 2018. URL https://aclanthology.info/papers/P18-1238/p18-1238.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL http://arxiv.org/abs/1706.03762.

Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. *CoRR*, abs/1411.5726, 2014. URL http://arxiv.org/abs/1411.5726.

Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. pp. 511–518, 2001.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144, 2016. URL http://arxiv.org/abs/1609.08144.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. *CoRR*, abs/1502.03044, 2015. URL http://arxiv.org/abs/1502.03044.

P Young, A Lai, M Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. 2:67–78, 01 2014.