Modulating and attending the source image during encoding improves Multimodal Translation

Jean-Benoit Delbrouck
University of Mons, Belgium
jean-benoit.delbrouck@umons.ac.be

Stéphane Dupont
University of Mons, Belgium
stephane.dupont@umons.ac.be

Abstract

We propose a new and fully end-to-end approach for multimodal translation where the source text encoder modulates the entire visual input processing using conditional batch normalization, in order to compute the most informative image features for our task. Additionally, we propose a new attention mechanism derived from this original idea, where the attention model for the visual input is conditioned on the source text encoder representations. In the paper, we detail our models as well as the image analysis pipeline. Finally, we report experimental results. They are, as far as we know, the new state of the art on three different test sets.

1 Introduction

Since the first multimodal machine translation (MMT) shared task [23] has been released, the community struggled to prove the effectiveness of images in the translation process. Most of the works [5, 18, 7, 11] naturally focused on using a soft attention mechanism [3] on the convolutional features (also called attention maps) of an image, alongside with a textual attention mechanism, because this approach has shown great success in image captioning [24]. First attempts were relatively unsuccessful (i.e. slightly lower than a strong monomodal baseline) and it was hard to figure out the real reasons of these underwhelming results. The last multimodal translation shared task [16] decided to address this issue by releasing two new test-sets containing pictures of new Flickr groups and sentences with ambiguous verbs so we know for sure the image could play a disambiguation role in the translation process. At the same time tough, the monomodal baseline got stronger and stronger with new findings regarding recurrent network architectures, such as layer normalization [2], making the improvements brought by a new modality thiner. The most successful recent try [6] focused on using the max-pooled features extracted from a CNN to modulate some components of the system (i.e. the target embeddings). So far, researchers extract the image features from a pre-trained CNN without any intervention from the encoder-decoder model used for translation.

We decide to take the leap and to modulate the feature extraction process by the linguistic input. More precisely, this paper aims to :

- Give a first try on a fully end-to end (visual and textual) multimodal translation model;
- Condition the forward pass of the CNN to extract visual features according to the textual encoder;
- Propose an encoder-based image attention model as opposed to the conventional attention mechanism used during decoding time;

In the area of NMT, two works are related to ours, in the sense that one modality analysis process is affected by the analysis of the other modality. Firstly, [15] proposed an architecture with an encoder shared between two decoders : one to output a translated sentence and one to reconstruct (imagine as the authors say) the image features. The encoder was thus trained to learn grounded representation.

Secondly, [12] used a grounded attention mechanism (referred as "pre-attention") where the image features were refined according to the encoder's representation of the source sentence.

2 Monomodal (Text-based) MT model

Our model is based on an encoder-decoder architecture with attention mechanism [3]. The encoder is a bi-directional RNN with Gated Recurrent Unit (GRU) layers [9, 8]. A forward RNN $\overrightarrow{f}_{\text{enc}}$ and a backward RNN $\overleftarrow{f}_{\text{enc}}$ both read an input sequence $x = (x_1, x_2, \ldots, x_M)$, ordered from x_1 to x_M and from x_M to x_1 respectively. Each RNN produces a hidden state $\widehat{\boldsymbol{h}}_i$ for each word x_i. We create a sequence of annotations $h = (\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_M)$, $\boldsymbol{h}_i = [\overrightarrow{\widehat{\boldsymbol{h}}}_i; \overleftarrow{\widehat{\boldsymbol{h}}}_i]$ where [ ] denotes the concatenation operation. Therefore, each annotation $\boldsymbol{h}_i$ now contains the summaries of both the preceding words and the following words.

The decoder f_{dec} is a CGRU (two stacked GRUs) that predicts the probability of a target sequence $y = (y_1, y_2, \ldots, y_K)$ based on h. At each decoding step t, an unnormalized attention score $\widehat{a}_i$ is computed for each source annotation $\boldsymbol{h}_i$ using the first GRU's hidden state $\boldsymbol{s}_t$ and $\boldsymbol{h}_i$ itself (equation 1):

$$\widehat{a}_i = \boldsymbol{V}_a^T \tanh(\boldsymbol{W}_h \boldsymbol{h}_i + \boldsymbol{W}_s \boldsymbol{s}_t) \quad (1) \qquad a_i = \frac{\widehat{a}_i}{\sum_m^M \widehat{a}_m} \quad (2) \qquad \boldsymbol{c}_t = \sum_i^M a_i \boldsymbol{h}_i \quad (3)$$

The attention vector $\boldsymbol{c}_t$ is calculated as a weighted average of the source states h as shown in equation 3. The second GRU computes the final state $\widehat{\boldsymbol{s}}$ of the decoder with $\boldsymbol{c}_t$ and $\boldsymbol{s}_t$. The decoder outputs a distribution over a vocabulary of fixed-size V based on the recurrent state of the second GRU $\widehat{\boldsymbol{s}}_t$, the previous words $y_{<t}$, and the attention vector $\boldsymbol{c}_t$:

$$\boldsymbol{o}_t = \tanh(\boldsymbol{y}_{t-1} + \boldsymbol{W}_{\widehat{s}} \widehat{\boldsymbol{s}}_t + \boldsymbol{W}_c \boldsymbol{c}_t) \quad (4) \qquad P(y_t | y_1, \ldots, y_{t-1}, x) = \text{softmax}(\boldsymbol{W}_o \boldsymbol{o}_t) \quad (5)$$

The whole model is trained end-to-end by minimizing the negative log likelihood of the target words using stochastic gradient descent.

3 Our multimodal MT model

As stated in the introduction, the convolutional network extracting the image features is now part of the training procedure. We chose a residual network (ResNet) who iteratively refines a representation by adding pass-through routing so that layers receive more detailed information rather than the information processed by the previous layer or adjacent to it. This modification enables to train deep convolutional networks without suffering too much from the vanishing gradient problem.

3.1 Residual Network

ResNets are built from residual blocks:

$$\boldsymbol{y} = \mathcal{F}(\boldsymbol{x}, \{\boldsymbol{W}_i\}) + \boldsymbol{W}_s \boldsymbol{x}$$

Here, $\boldsymbol{x}$ and $\boldsymbol{y}$ are the input and output vectors of the layers considered. The function $\mathcal{F}(\boldsymbol{x}, \{\boldsymbol{W}_i\})$ is the residual mapping to be learned. For an example, if we consider two layers, $\mathcal{F} = \boldsymbol{W}_2 \sigma(\boldsymbol{W}_1 \boldsymbol{x})$ where σ denotes ReLu function. The operation $\mathcal{F} + \boldsymbol{x}$ is the shortcut connection and consists of an element-wise addition. Therefore, the dimensions of $\boldsymbol{x}$ and $\mathcal{F}$ must be equal. When this is not the case (e.g., when changing the input/output channels), the $\boldsymbol{W}_s$ matrix performs a linear projection by the shortcut connections to match the dimension. Finally, it performs a last second nonlinearity after the addition (i.e., $\sigma(y)$). A group of blocks are stacked to form a stage of computation. The general ResNet architecture starts with a single convolutional layer followed by 4 stages of computation.

3.2 Conditional Batch Normalization

A ResNet adopts batch normalization (BN)[19] right after each convolution and before activation. This techniques tackles the problem of internal covariate shift (the distribution of each layer's inputs changes during training, as the parameters of the previous layers change) and addresses it by normalizing layer inputs :

$$\widehat{\boldsymbol{x}}^{(k)} = \frac{\boldsymbol{x}^{(k)} - \mathrm{E}_B[\boldsymbol{x}^{(k)}]}{\sqrt{\mathrm{Var}_B[\boldsymbol{x}^{(k)}] + \epsilon}} \quad (6) \qquad \boldsymbol{y}^{(k)} = \gamma^{(k)}\widehat{\boldsymbol{x}}^{(k)} + \beta^{(k)} \quad (7)$$

The network applies the above equation 6 to make each feature dimension k of the input $\boldsymbol{x}$ in the whole mini-batch follow a zero mean and unit variance Gaussian. On top of that, the model has the opportunity to shift and scale the result as shown in equation 7 before going through the the non-linearity (ReLu). At inference time, the batch mean and variance are replaced by a single empirical mean and variance of activations during training.

To modulate the visual processing by language, we will predict a small change in the shift and scale parameters of equation 7 according to the text-based source annotations sequence h as already been proposed in the related VQA task [10] (called "*Modulated ResNet*" in the author's paper). We call this conditional batch normalization. To do so, we use a one-hidden-layer MLP to predict these deltas for all feature maps within the layer:

$$\Delta\gamma, \Delta\beta = \mathrm{MLP}(q(h)) \quad (8) \qquad \widehat{\gamma}^{(k)} = \gamma^{(k)} + \Delta\gamma^{(k)} \quad (9) \qquad \widehat{\beta}^{(k)} = \beta^{(k)} + \Delta\beta^{(k)} \quad (10)$$

where $q(\{\boldsymbol{h}_1, \ldots, \boldsymbol{h}_M\}) = \tanh\left(\boldsymbol{W}_q \cdot \frac{1}{M}\sum_{i=1}^{M} \boldsymbol{h}_i\right)$

3.3 Image Features

This section aims to explain which image features are extracted from the ResNet and how the model described in section 2 uses it. Commonly, two types of features are useful for machine translation: global pooled features (a vector of features) and convolutional features (also called an attention map, a 3D-matrice). Because we use one or the other, our model now has two variants referred as "*pool5*" and "*conv*" in the results section.

3.3.1 Global pool5 Features

In the ResNet architecture, at the end of the 4th stage sits a max-pooling layer just before the fully connected layer whose output is a global 2048-dimensional visual representation V of the image. We use V to modulates each source annotation $\boldsymbol{h}_i$ using element-wise multiplication (as done in [6]):

$$\boldsymbol{h}_i = \boldsymbol{h}_i \odot \tanh(\boldsymbol{W}_{\text{pool}} \cdot V) \quad (11)$$

Because V is a vector of features, a "*pool5*" model does not need a second attention mechanism.

3.3.2 Convolutional Features

At the end of the ResNet 3rd stage, after the ReLu activation (res4f), we extract convolutional feature maps of 7x7x1024 (the 3D matrice) that are regarded as 49 spatial annotations of 1024-dimension each. We use a soft attention mechanism over the 49 visual spatial locations $(\boldsymbol{l}_1, \ldots, \boldsymbol{l}_{49})$ at each decoding step t. It is the exact same mechanism of section 2 but with $\boldsymbol{h}_i$ replaced by $\boldsymbol{l}_i$:

$$\widehat{a}_i = \boldsymbol{V}_a^T \tanh(\boldsymbol{W}_l \boldsymbol{l}_i + \boldsymbol{W}_s \boldsymbol{s}_t) \quad (12) \qquad a_i = \frac{\widehat{a}_i}{\sum_m^{49} \widehat{a}_m} \quad (13) \qquad V_t = \sum_i^{49} a_i \boldsymbol{l}_i \quad (14)$$

This hence constitutes an additional attention mechanism to the one described in section 2, applied to the visual annotations $\boldsymbol{l}_i$ rather than the text annotations $\boldsymbol{h}_i$. The weighted sum of the attention process of equation 14 leaves us with a 1024-dimensional visual representation V of the image. We then use it to modulates each source annotation $\boldsymbol{h}_i$ as described in equation 11.

4 Encoder-based image attention

In machine translation, any image attention mechanism on convolutional features – soft [3], local or stochastic [11] – happens on the decoder-side (based on $\boldsymbol{s}_t$ as seen in the previous section 3.3.2). At each time-step t, the decoder has to decide which spatial features are interesting to decode the next translated token. However, when it comes to translate a sentence in real life, we rather tend to imagine a visual representation as soon as we read the source sentence. The encoder should probably be the strongest place to build a strong visual representations for our translation task. This hypothesis is reinforced by the additional role the encoder now endorse: modulating the visual processing as described in section 3.2. Because the encoder now plays a part in the making of these convolutional features, we propose to apply the attention mechanism for the visual representation during the encoding.

As shown in Figure 1, the encoder now builds, at every time-step i, a textual representation $\boldsymbol{h}_i$ and a visual representation V_i of the word x_i. The encoder visual attention module is similar to the one described in subsection 3.3.2, but takes place in the encoder. Therefore, equation 12 does not depend on the decoder state $\boldsymbol{s}_t$ anymore but on the source annotation $\boldsymbol{h}_i$. Now the decoder, at every decoding-step t, still computes a soft alignment over the source sentence annotations $\boldsymbol{h}$ and hence gets, on its way, the visual representations V as well. In contrast to earlier works on multimodal MT, the decoder is now equipped with only one multimodal attention mechanism, as the textual and visual representation of a word are computed in the encoder.

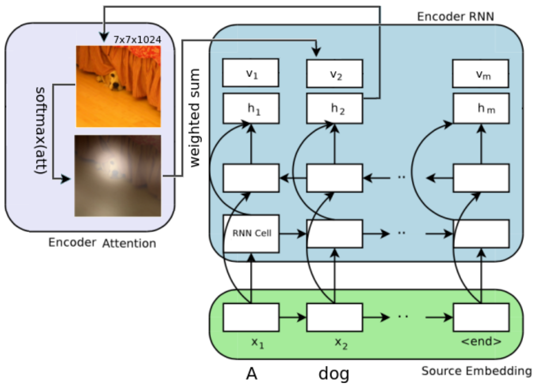

Figure 1: Encoder-based attention for time-step 2

5 Dataset

We used the Multi30K dataset [14] which is an extended version of the Flickr30K Entities. For each image, one of the English descriptions was selected and manually translated into German by a professional translator. As training and development data, 29,000 and 1,014 triples are used respectively. We dispose of three test sets to score our models. The flickr Test2016 and the Flickr Test2017 set contain 1000 image-caption pairs and the ambiguous MSCOCO test set [16] 461 pairs.

6 Experiments and Results

Previous work [12] showed that using visual features from different CNNs lead to variable translation performance for a same encoder-decoder model. In the this paper, we stick to two different versions of ResNet : the ResNet v1 detailed in our section 3.2 and ResNet v2, a slight variant of ResNet v1 as described in [17]. The image preprocessing operation is described in Appendix A.

Table 1: RN = ResNet, CBN = Conditional Batch Normalization, FT = fine tuning of the last ResNet stage and (*) is used as baseline

Model	**Test 2016**		**Test 2017**		**Ambiguous COCO**	
	BLEU↑	METEOR↑	BLEU↑	METEOR↑	BLEU↑	METEOR↑
Pre-trained Pool5* [6]	38.4 ± 0.3	57.8 ± 0.5	31.1 ± 0.7	51.9 ± 0.2	27.0 ± 0.7	47.1 ± 0.7
RN v1 CBN Conv	38.9 ± 0.3	57.1 ± 0.6	30.0 ± 1.1	50.9 ± 0.2	26.3 ± 0.9	46.5 ± 0.6
RN v1 CBN Pool5	39.4± 0.8	**57.9** ± 0.6	**31.5** ± 0.4	52.2 ± 0.5	**27.4** ± 0.9	48.1 ± 0.6
RN v2 CBN Pool5	38.7 ± 0.3	56.5 ± 0.5	30.1 ± 0.7	51.1 ± 0.6	26.5 ± 0.7	46.3 ± 0.6
RN v1 CBN FT Pool5	38.2 ± 0.6	57.5 ± 0.6	30.4 ± 0.6	51.4 ± 0.7	26.4 ± 0.9	46.8 ± 0.4
RN v1 CBN enc-att	**40.5** ± 0.8	**57.9** ± 0.6	31.4 ± 0.4	**52.5** ± 0.7	27.3 ± 0.9	**48.5** ± 0.4

Both ResNets are pretrained on ImageNet. Unless stated otherwise, ResNet parameters are frozen during training, including scalars $\gamma^{(k)}$ and $\beta^{(k)}$ from section 3.2. We use the metrics BLEU [21] and METEOR [13] to evaluate the quality of our models' translations. We stop a training if there is no METEOR improvements on the dev-set for 10k steps.

First and foremost, we can notice that conditional batch normalization enhances our model translations, specifically when using the global features (*RN v1 CBN Pool5*). Applying CBN at every ResNet stage lead to the best improvement (cfr. table 4 in Appendix C) but we also find that fine-tuning the last layer does not improve this performance (*RN v1 CBN FT Pool5*). This result reinforce our main postulate that modulating the visual process by language enhance the quality of the translations.

Secondly, when using decoder-based attention on convolutional features as described in section 3.3.2, the model performs poorly (*RN v1 CBN Conv*). As stated in the introduction, it's not sure we have enough data to successfully train an attention model. Nevertheless, using the encoder-based attention (section 4) palliates this gap. Indeed, both models *RN v1 CBN enc-att* and *RN v1 CBN Conv* have very close results.

Lastly, using a ResNet v2 slightly deteriorates the results. The key difference between the two architectures is the use of batch normalization before every weight layer. A more in-depth study of the model parameters and architecture might be needed to figure out the cause of this small drop. Another possible future work would be the use larger images as ResNet inputs (448x448) to enjoy convolutional features of 196 spatial locations, as this has shown great success in VQA.

7 Acknowledgements

This work was partly supported by the Chist-Era project IGLU with contribution from the Belgian Fonds de la Recherche Scientique (FNRS), contract no. R.50.11.15.F, and by the FSO project VCYCLE with contribution from the Belgian Waloon Region, contract no. 1510501.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. 2016. URL https://arxiv.org/pdf/1603.04467.pdf.

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. In *Neural Information Processing System Deep Learning Symposium 2017*, 2016. URL https://arxiv.org/pdf/1607.06450.pdf.

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015. URL https://arxiv.org/pdf/1409.0473.pdf.

[4] Denny Britz, Anna Goldie, Thang Luong, and Quoc Le. Massive Exploration of Neural Machine Translation Architectures. *ArXiv e-prints*, March 2017.

[5] Ozan Caglayan, Walid Aransa, Yaxing Wang, Marc Masana, Mercedes García-Martínez, Fethi Bougares, Loïc Barrault, and Joost van de Weijer. Does multimodality help human and machine for translation and image captioning? In *Proceedings of the First Conference on Machine Translation*, pages 627–633, Berlin, Germany, August 2016. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W/W16/W16-2358.

[6] Ozan Caglayan, Walid Aransa, Adrien Bardet, Mercedes García-Martínez, Fethi Bougares, Loïc Barrault, Marc Masana, Luis Herranz, and Joost van de Weijer. Lium-cvc submissions for wmt17 multimodal translation task. In *Proceedings of the Second Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 432–439, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W17-4746.

[7] Iacer Calixto, Desmond Elliott, and Stella Frank. Dcu-uva multimodal mt system report. In *Proceedings of the First Conference on Machine Translation*, pages 634–638, Berlin, Germany,

August 2016. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W/W16/W16-2359.

[8] Kyunghyun Cho, Bart van Merriënboer, Çağlar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D14-1179.

[9] Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. *Empirical evaluation of gated recurrent neural networks on sequence modeling*. 2014.

[10] Harm de Vries, Florian Strub, Jérémie Mary, Hugo Larochelle, Olivier Pietquin, and Aaron Courville. Modulating early visual processing by language. 2017. URL https://arxiv.org/pdf/1707.00683.pdf.

[11] Jean-Benoit Delbrouck and Stéphane Dupont. An empirical study on the effectiveness of images in multimodal neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 921–930, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D17-1096.

[12] Jean-Benoit Delbrouck and Stephane Dupont. Multimodal compact bilinear pooling for multimodal neural machine translation. *arXiv preprint arXiv:1703.08084*, 2017. URL https://arxiv.org/pdf/1703.08084.pdf.

[13] Michael Denkowski and Alon Lavie. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the EACL 2014 Workshop on Statistical Machine Translation*, 2014.

[14] D. Elliott, S. Frank, K. Sima'an, and L. Specia. Multi30k: Multilingual english-german image descriptions. pages 70–74, 2016.

[15] Desmond Elliott and Ákos Kádár. Imagination improves multimodal translation. *CoRR*, abs/1705.04350, 2017. URL http://arxiv.org/abs/1705.04350.

[16] Desmond Elliott, Stella Frank, Loïc Barrault, Fethi Bougares, and Lucia Specia. Findings of the Second Shared Task on Multimodal Machine Translation and Multilingual Image Description. In *Proceedings of the Second Conference on Machine Translation*, Copenhagen, Denmark, September 2017.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. *Identity Mappings in Deep Residual Networks*, pages 630–645. Springer International Publishing, Cham, 2016. ISBN 978-3-319-46493-0. doi: 10.1007/978-3-319-46493-0_38. URL https://doi.org/10.1007/978-3-319-46493-0_38.

[18] Po-Yao Huang, Frederick Liu, Sz-Rung Shiang, Jean Oh, and Chris Dyer. Attention-based multimodal neural machine translation. In *Proceedings of the First Conference on Machine Translation*, pages 639–645, Berlin, Germany, August 2016. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W/W16/W16-2360.

[19] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *JMLR Proceedings*, pages 448–456. JMLR.org, 2015.

[20] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 177–180, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1557769.1557821.

[21] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL `http://dx.doi.org/10.3115/1073083.1073135`.

[22] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural Machine Translation of Rare Words with Subword Units. In *In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2016. URL `http://www.aclweb.org/anthology/P16-1162`.

[23] Lucia Specia, Stella Frank, Khalil Sima'an, and Desmond Elliott. A shared task on multimodal machine translation and crosslingual image description. In *Proceedings of the First Conference on Machine Translation*, pages 543–553, Berlin, Germany, August 2016. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/W/W16/W16-2346`.

[24] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In David Blei and Francis Bach, editors, *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 2048–2057. JMLR Workshop and Conference Proceedings, 2015. URL `http://jmlr.org/proceedings/papers/v37/xuc15.pdf`.

A ResNets

A.1 Preprocessing

ResNet v1 has been trained on ImageNet with a vgg preprocessing. It consists of a random crop and a random flip. ResNet v2 applies the inception preprocessing that uses, on top of the vgg preprocessing, random color distortion (hue, contrast, brightness and saturation). Input image size for ResNet v1 and v2 are respectively of 224x224x3 and 299x299x3.

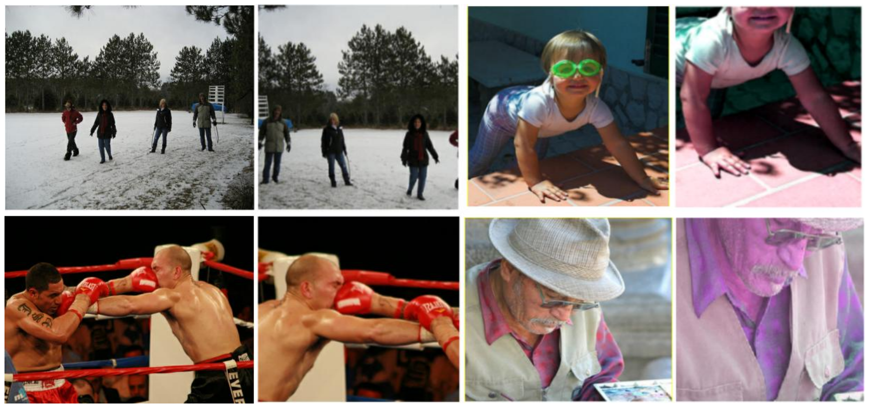

Figure 2: Top-left and bottom-left are the VGG processing. Top right and bottom right are the Inception processing

A.2 Parameters

Table 2: ResNets Parameters

Parameter	Value
ResNet v1 version	ResNet-50
ResNet v2 version	ResNet-50
ResNet v1 input size	224x224x3
ResNet v2 input size	299x299x3
ResNet v1 CBN inference	moving average
ResNet v2 CBN inference	exponential moving average
CBN decay	0.99
CBN damping factor ϵ	1e-5
CBN MLP hidden units	512
Blocks with CBN	all (by default)

B Sequence to sequence model

To conduct our experiments, we use the TensorFlow [1] library as well as the google seq2seq framework [4]. We release our code on github [1]. We normalize and tokenize English and German descriptions using the Moses tokenizer scripts [20]. We use the byte pair encoding algorithm on the train set to convert space-separated tokens into subwords [22] with 10K merge operation, reducing our vocabulary size to 5234 and 7052 words for English and German respectively. Embeddings are learned along with the model.

B.1 Encoder

Both encoders are equipped with layer normalization [2] where each hidden unit adaptively normalizes its incoming activations with a learnable gain and bias.

B.2 Decoder

We initialize the decoder hidden state $\boldsymbol{h}_0$ of the CGRU with a non-linear transformation of the average source annotation:

$$\boldsymbol{h}_0 = \tanh\left(\boldsymbol{W}_{init} \cdot \frac{1}{M}\sum_{i=1}^{M}\boldsymbol{h}_i\right), \boldsymbol{h}_i \in h \tag{15}$$

The decoder is a conditional GRU [2] that consists of two stacked GRU activations called REC$_1$ and REC$_2$ and an attention mechanism f_{att} in between (called ATT in the footnote paper). At each time-step t, REC1 firstly computes a hidden state proposal $\boldsymbol{s}_t$ based on the previous hidden state $\widehat{\boldsymbol{s}}_{t-1}$ and the previous emitted word y_{t-1}:

$$\begin{aligned}
\boldsymbol{z}_t &= \sigma\left(\boldsymbol{W}_z\boldsymbol{E}_Y[y_{t-1}] + \boldsymbol{U}_z\widehat{\boldsymbol{s}}_{t-1}\right)\\
\boldsymbol{r}_t &= \sigma\left(\boldsymbol{W}_r\boldsymbol{E}_Y[y_{t-1}] + \boldsymbol{U}_r\widehat{\boldsymbol{s}}_{t-1}\right)\\
\underline{\boldsymbol{s}}_t &= \tanh\left(\boldsymbol{W}\boldsymbol{E}_Y[y_{t-1}] + \boldsymbol{r}_t \odot (\boldsymbol{U}\widehat{\boldsymbol{s}}_{t-1})\right)\\
\boldsymbol{s}_t &= (1-\boldsymbol{z}_t) \odot \underline{\boldsymbol{s}}_t + \boldsymbol{z}_t \odot \widehat{\boldsymbol{s}}_{t-1}
\end{aligned} \tag{16}$$

Then, the attention mechanism computes $\boldsymbol{c}_t$ over the source sentence using the annotations sequence h and the intermediate hidden state proposal $\boldsymbol{s}_t$ (cfr. section 2).

[1] `https://github.com/jbdel/mmt_cbn`

[2] `https://github.com/nyu-dl/dl4mt-tutorial/blob/master/docs/cgru.pdf`

Finally, the second recurrent cell REC$_2$, computes the hidden state $\widehat{\boldsymbol{s}}_t$ of the cGRU by looking at the intermediate representation $\boldsymbol{s}_t$ and context vector $\boldsymbol{c}_t$:

$$\begin{aligned}
\boldsymbol{z}_t =& \sigma\left(\boldsymbol{W}_z \boldsymbol{c}_t + \boldsymbol{U}_z \boldsymbol{s}_t\right) \\
\boldsymbol{r}_t =& \sigma\left(\boldsymbol{W}_r \boldsymbol{c}_t + \boldsymbol{U}_r \boldsymbol{s}_t\right) \\
\widehat{\underline{\boldsymbol{s}}}_t =& \tanh\left(\boldsymbol{W} \boldsymbol{c}_t + \boldsymbol{r}_t \odot (\boldsymbol{U} \boldsymbol{s}_t)\right) \\
\widehat{\boldsymbol{s}}_t =& (1 - \boldsymbol{z}_t) \odot \underline{\boldsymbol{s}}_t + \boldsymbol{z}_t \odot \boldsymbol{s}_t
\end{aligned} \tag{17}$$

B.3 Parameters

Table 3: Sequence to sequence parameters

Parameter	Value
Source and target embeddings	128
GRU and CGRU Layer size	256
Attention size	256
GRU input dropout	0.7
GRU output dropout	0.5
CGRU input dropout	1.0
CGRU output dropout	1.0
Softmax output dropout 4	0.5
Optimizer	Adam
Learning rate	0.0004
Optimize epsilon	0.0000008
Batch-size	32
Inference Beam-Size	12

C Further Results

Table 4: Use of CBN in the different ResNet stage (Stage 4 only still to compute for camera ready version)

RN v1 CBN Pool5	**Test 2016**		**Test 2017**		**Ambiguous COCO**	
	BLEU↑	METEOR↑	BLEU↑	METEOR↑	BLEU↑	METEOR↑
All	39.4$\pm$ 0.8	57.9 $\pm$ 0.6	31.5 $\pm$ 0.4	52.2 $\pm$ 0.5	27.4 $\pm$ 0.9	48.1 $\pm$ 0.6
Stages 2 - 4	38.7 $\pm$ 0.6	57.0 $\pm$ 0.7	31.4$\pm$ 0.9	51.4 $\pm$ 0.4	27.3 $\pm$ 0.9	46.8 $\pm$ 0.9
Stages 3 - 4	38.8 $\pm$ 0.8	56.1 $\pm$ 0.7	30.4 $\pm$ 0.8	51.1 $\pm$ 0.6	25.9 $\pm$ 1.0	46.0 $\pm$ 0.6