## A Models Set-Up

**General Settings** We used the implementation of *Nematus* (Sennrich et al., 2017) for both models. We trained each architecture (i.e., GRU and Transformer) three times. For testing, we ensembled the settings which obtained the best results in the development sets in each training execution for GRUs, whereas for the Transformer, we selected the setting which obtained the best result in the respective development set.

Models were trained using stochastic gradient descent with Adam (Kingma and Ba, 2015) ($\beta_1 = 0.9$, $\beta_2 = 0.98$, $\epsilon = 10^{-9}$) for a maximum of 200,000 updates. They were evaluated on the development sets after every 5,000 updates and early stopping was applied with patience 30 based on cross-entropy. Encoder, decoder and softmax embeddings were tied, whereas decoding was performed with beam search of size 5 to predict sequences with length up to 100 tokens.

**GRU Settings** Bidirectional GRUs with attention were used as described in Sennrich et al. (2017). Source and target word embeddings were 300D each, whereas hidden units were 512D. We applied layer normalization as well as dropout with a probability of 0.1 in both source and target word embeddings and 0.2 for hidden units.

**Transformer Settings** Both encoder and decoder consisted of $N = 6$ identical layers. Word embeddings and hidden units were 512D each, whereas the inner dimension of feed-forward sublayers were 2048D. The multi-head attention sublayers consisted of 8 heads each. Dropout of 0.1 were applied to the sums of word embeddings and positional encodings, to residual connections, to the feed-forward sub-layers and to attention weights. At training, models had 8000 warm-up steps and label smoothing of 0.1.

**Word Segmentation** In the lexicalization step of the pipeline and in the end-to-end architecture, *byte-pair encoding* (BPE) (Sennrich et al., 2016) was used to segment the tokens of the target template and text, respectively. The model was trained to learn 20,000 merge operations with a threshold of 50 occurrences.

**NeuralREG** To generate referring expressions in the pipeline architecture, we used the concatenative-attention version of the NeuralREG algorithm (Castro Ferreira et al., 2018). We follow most of the settings in the original paper, except for the number of training epochs, mini-batches, dropout, beam search and early stop of the neural networks, which we respectively set to 60, 80, 0.2, 5 and 10. Another difference is in the input of the model: while NeuralREG in the original paper generates referring expressions based on templates where only the references are delexicalized, here the algorithm generates referring expressions based on a template where verbs and determiners are also delexicalized as previously explained.

## References

Thiago Castro Ferreira, Diego Moussallem, Ákos Kádár, Sander Wubben, and Emiel Krahmer. 2018. NeuralREG: An end-to-end approach to referring expression generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1959–1969. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Rico Sennrich, Orhan Firat, Kyunghyun Cho, Alexandra Birch, Barry Haddow, Julian Hitschler, Marcin Junczys-Dowmunt, Samuel Läubli, Antonio Valerio Miceli Barone, Jozef Mokry, and Maria Nadejde. 2017. Nematus: a toolkit for neural machine translation. In *Proceedings of the Software Demonstrations of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 65–68, Valencia, Spain. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL'16, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.