Neural Speech Translation using Lattice Transformations and Graph Networks - Supplementary Material

A Model Hyperparameters

Our implementation is based on the Sockeye toolkit for Neural Machine Translation (Hieber et al., 2017). Besides the specific hyperparameter values mentioned in the paper, all other hyperparameters are set to the default values in Sockeye. We detail them here for completeness:

A.1 Model

- The baseline encoder use a BiLSTM followed by a unidirectional LSTM. The decoder in all models use a 2-layer LSTM.
- The attention module uses a bilinear scoring function (*general* as in Luong et al. (2015)).
- The max sequence length during training is 200. This is because lattices are in general much larger than 1-best inputs.
- All dimensionalities are fixed to 512.
- For the GGNN models, we use 8 layers and ReLU as the activation function.

A.2 Training

- We use 16 as the batch size. This is in line with recent evidence showing that smaller batch sizes lead to better generalisation performance (Keskar et al., 2017). The drawback is that smaller batches makes training time slower. However, this was not a problem in our experiments due to the medium size of the datasets.
- Bucketing is used to speed up training: we use 10 as the bucket size.
- Models are trained using cross-entropy as the loss.
- We save parameter checkpoints at every full epoch on the training set.

- We use early stopping by perplexity on the dev set with patience 8 (training stops if dev perplexity do not improve for 8 checkpoints).
- A maximum of 30 epochs/checkpoints is used. All our models stopped training before reaching this limit.
- We use 0.5 dropout on the input embeddings, before they are fed to the encoder.
- Weigths are initalised using Xavier initialisation (Glorot and Bengio, 2010), with except of forget biases in the LSTMs which are initialised by 0.
- We use Adam (Kingma and Ba, 2015) as the optimiser with 0.0003 as the initial learning rate.
- Learning rate is halved every time dev perplexity does not improve for 3 epochs/checkpoints.
- Gradient clipping is set to 1.0.

A.3 Decoding

- We use beam search to decode, using 5 as the beam size.
- Ensembles are created by averaging log probabilities at every step (*linear* in Sockeye). Attention scores are averaged over the 5 models at every step.

References

Glorot, X. and Bengio, Y. (2010). Understanding the Difficulty of Training Deep Feedforward Neural Networks. In *Proceedings of AISTATS*, volume 9, pages 249–256.

Hieber, F., Domhan, T., Denkowski, M., Vilar, D., Sokolov, A., Clifton, A., and Post, M. (2017). Sockeye: A Toolkit for Neural Machine Translation. *arXiv preprint*, pages 1–18.

Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., and Tang, P. T. P. (2017). On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima. In *Proceedings of ICLR*, pages 1–16.

Kingma, D. P. and Ba, J. (2015). Adam: A Method for Stochastic Optimization. In *Proceedings of ICLR*, pages 1–15.

Luong, M.-T., Pham, H., and Manning, C. D. (2015). Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of EMNLP*, pages 1412–1421.