Learning Hidden Unit Contribution for Adapting Neural Machine Translation Models

David Vilar
Amazon Research
Berlin, Germany
dvilar@amazon.com

Abstract

In this paper we explore the use of *Learning Hidden Unit Contribution* for the task of neural machine translation. The method was initially proposed in the context of speech recognition for adapting a general system to the specific acoustic characteristics of each speaker. Similar in spirit, in a machine translation framework we want to adapt a general system to a specific domain. We show that the proposed method achieves improvements of up to 2.6 BLEU points over a general system, and up to 6 BLEU points if the initial system has only been trained on out-of-domain data, a situation which may easily happen in practice. The good performance together with its short training time and small memory footprint make it a very attractive solution for domain adaptation.

1 Introduction

Domain adaptation for neural machine translation (NMT) is starting to get more attention from the scientific community. Often researchers and machine translation practitioners want to improve the performance of their systems on a domain for which they were not explicitly optimized. Due to the high training times needed to develop NMT systems, often up to several weeks, efficient methods for improving an existing system for a specific domain can have important practical applications.

In this paper we review "Learning Hidden Unit Contribution" (§ 3), a method developed initially for speaker adaptation in speech recognition systems, and apply it to the task of neural machine translation (§§ 4 and 5). We show that it improves translation quality on in-domain data, while at the same time keeping the translation quality of out-of-domain data intact (§ 6). Due to its small memory footprint and short training time it can be realistically applied to adapt large, general domain systems in order to improve their performance on specific domains.

2 Neural Machine Translation

In this section we will provide a short overview of NMT. For a more detailed description the reader is referred to existing literature. An NMT system is mainly composed of two parts. The first one, called the encoder, produces a sequence of vectors which constitute a representation of the input sentence in a continuous vector space. The decoder takes this sequence of vectors and generates a new sequence of words in the target language, which corresponds to the translation of the given sentence. An additional attention mechanism helps guiding the translation process.

Most NMT systems are based on recurrent networks, usually LSTMs or GRUs (Bahdanau et al., 2014), although recently new approaches to neural machine translation have been proposed (Vaswani et al., 2017; Gehring et al., 2017) which are not based on recurrent networks, but keep the encoder-decoder structure. All the approaches described in this paper are equally applicable to these models.

3 Learning Hidden Unit Contribution

Learning Hidden Unit Contribution (LHUC) is a method first introduced by Swietojanski and Renals (2014) and Swietojanski et al. (2016) for speaker adaptation in neural speech recognition systems. The goal of speaker adaptation is to tailor an existing speech recognition system to the specific acoustic characteristics of a given speaker. In normal conditions, the amount of training data available for one speaker is rather limited; therefore, the authors' goal was to develop a method that would be able to adapt with a small sample size. Additionally, when adapting a system, there is the danger that the system's performance on the

Proceedings of NAACL-HLT 2018, pages 500–505
New Orleans, Louisiana, June 1 - 6, 2018.

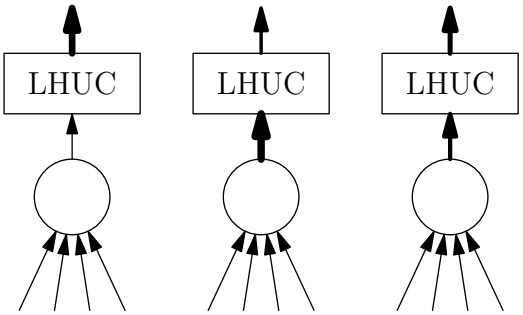

Figure 1: Illustration of the LHUC approach. Three units of a hidden layer are depicted. Each unit has an additional component that can scale the value of the original output. The number of additional parameters is linear in the number of hidden units (see Eq. 2).

general domain degrades significantly due to overfitting, what in neural network literature is sometimes called "catastrophic forgetting" (McCloskey and Cohen, 1989; Ratcliff, 1990).

The intuition behind LHUC is that different network units specialize on different aspects of the task, and thus, when shifting domain the importance of each unit may change from the original domain on which the system was trained. An example of this behaviour (unrelated to translation) is shown by Radford et al. (2017). They trained a character-level language model on product review texts and found out that one specific neuron provided a quite accurate representation of the sentiment (positive or negative) of the text. While this neuron can provide valuable information for this task, it may not be so relevant for other domains where sentiment is not as important (e.g. news). At the same time, other neurons may become more important (e.g. ensuring a more formal style of the text).

LHUC introduces an additional multiplicative amplitude element to the output of each hidden unit in the network. As such the contribution of the hidden unit can be amplified (values greater than 1) for the units that are more relevant to the task, or dampened (values close to 0) for units that are not as important. The approach is illustrated in Figure 1. Coming back to the sentiment neuron example above, for tasks where sentiment is important (product reviews), LHUC will assign a high weight to the sentiment neuron, while for other tasks (news), the corresponding weight will be low. The weights are learned automatically from the available in-domain data.

More formally, let

$$\mathbf{h}^{(l)} = \phi_\theta(\mathbf{h}^{(l-1)}, \dots) \tag{1}$$

be a general equation for the output of a layer in a neural network, parametrized by the set θ. In case of a feed-forward layer, ϕ would be an affine transformation followed by a non-linearity, θ would be the parameter matrix of the linear transformation and the bias vector, and no additional arguments would be required. Other models like LSTM layers (Hochreiter and Schmidhuber, 1997) have a more complex structure and a memory state as an additional argument.

As the LHUC method is very general, the actual form of the layer activations does not need to be specified in complete detail; it is only important to note that it produces a vector, and is dependent of the previous layer. LHUC modifies the activation function by introducing a multplicative element

$$\mathbf{h}^{(l)}_{\text{LHUC}} = a(\boldsymbol{\rho}^{(l)}) \circ \phi_\theta(\mathbf{h}^{(l-1)}_{\text{LHUC}}, \dots) \tag{2}$$

where $\boldsymbol{\rho}^{(l)}$ is a layer-dependent vector of new parameters (independent of θ), of the same dimension as $\mathbf{h}^{(l)}$, and $\circ$ denotes element-wise vector multiplication. The $a(\cdot)$ function is a scaled element-wise sigmoid function. The range of the scaled sigmoid is limited to the interval $[0, 2]$[1]

$$a(\mathbf{x}) = \frac{2}{1 + e^{-\mathbf{x}}}\,. \tag{3}$$

The parameter vectors $\boldsymbol{\rho}^{(l)}$ are trained using standard backpropagation, keeping the initial parameter set θ constant during the LHUC training pass. As can be seen, the number of parameters grows only linearly in the number of units in each layer, instead as the usual quadratic growth for the number of parameters in most neural models.

4 Domain Adaptation for NMT

Domain adaptation for phrase-based and related approaches to machine translation has been extensively investigated, e.g. (Schwenk, 2008; Axelrod et al., 2011; Carpuat et al., 2013). Neural machine translation, being a relatively new approach to MT, has not seen so many works going into this direction yet. Previous methods can of course still be applied as long as they are model-independent, e.g. data selection methods as shown

[1] The value of 2 is chosen as to be able to amplify the value (it can be doubled) but without overshadowing the value of the other units, which could happen with bigger values.

in (van der Wees et al., 2017) or self-training approaches like (Bertoldi and Federico, 2009), of which back-translation (Sennrich et al., 2015) can be considered a special case.

Specific for neural machine translation, Freitag and Al-Onaizan (2016) present a really simple method, where the parameters of an already trained system are taken as the starting point of another training run using only in-domain data. This simple method achieves good results and has the advantage that no additional implementation work has to be carried out. Chu et al. (2017) propose a refinement of this method where the in-domain data is mixed with the out-of-domain data.

Chen et al. (2017) propose to use a domain classifier to weight the cost of the training data differently according to the similarity to the in-domain data, in what can be considered a tighter integration of previous data-selection methods.

Sennrich et al. (2016) introduce a simple method for controlling the politeness of an NMT system, which can also be used for domain adaptation. They add a special tag to the source sentence denoting the politeness level of the source sentence and the system is able to use this information to improve the translation output. This technique can be extended to domain adaptation by combining the text of the different domains and marking them with a specific, domain-dependent tag. Johnson et al. (2016) use this technique in an "extreme" domain adaptation setting, where the domains are actually different languages.

5 LHUC for Domain Adaptation of NMT

In this paper we propose to use LHUC for domain adaptation of NMT systems. For this, we first train a system on the general domain data. Once such a system is available, we add the LHUC component and adapt it for the characteristics of the domain, similar to the adaptation to the acoustic characteristics of a speaker in the case of speech recognition.

We will explore two different scenarios, in the first one all the data, both in-domain and out-of-domain, is available from the beginning. As such the initial system already has seen in-domain examples at training time and the domain adaptation step is mainly a "fine tuning" step.

In the second scenario we will assume that the initial system has been trained only on out-of-domain data. This system will then be adapted for the new in-domain data, which has not been seen at the initial training stage. This scenario has an important role in practical applications, e.g. a general domain system has been trained and tuned to offer general domain translation, but it needs to be adapted to specific domains in order to provide better quality. In some cases, the time necessary for adapting such a system may also play a critical role on the applicability of the method.

LHUC provides an elegant solution for domain adaptation. Due to its reduced number of parameters it can be trained in a much shorter time than a full system. Furthermore, because it is an "add-on" for an already existing system it can be activated or de-activated on-demand. This effectively solves the catastrophic forgetting effect found in other adaptation techniques, as the general system can still be accessed at any time.

Corpus	Sents	Words	Voc
WMT	5M	135M/141M	1.8M/877K
IWSLT	197K	3.7M/4M	122K/55K

Table 1: Training corpora statistics. "Sents" denotes number of sentences, "Words" refers to number of running words after tokenization and "Voc" is the size of the vocabulary, e.g. the number of unique words in the corresponding corpus.

6 Experimental Results

Similar to Freitag and Al-Onaizan (2016), we present results on the IWSLT 2016[2] German to English TED dataset (Cettolo et al., 2016), consisting of transcribed and translated TED talks. As out-of-domain data we use the same year's WMT data (Bojar et al., 2016). We report results on the TED 2013 and TED 2014 (the newest ones with provided references) and additionally on the newstest 2016 dataset for measuring the performance on out-of-domain data. Statistics for the training corpora are given in Table 1. It can be seen that the WMT data (out-of-domain) is an order of magnitude bigger that the in-domain IWSLT data.

Our system is a recurrent encoder-decoder NMT model, with one bidirectional LSTM layer with 1024 units in the encoder and one layer with 1024 units in the decoder. The data has been BPE-encoded using 32K merge operations, and the em-

[2]Freitag and Al-Onaizan (2016) used an older version of the corpora.

bedding layer has a dimension of 512. Training has been performed with the Adam algorithm (Kingma and Ba, 2014). The provided TED dev set was used as stopping criterion (or newstest14 for the case of a WMT-only system). Experiments have been carried out using Sockeye (Hieber et al., 2017), and the LHUC code has been open sourced as part of it.

For LHUC experiments, both the encoder and decoder hidden units have been expanded with the additional scaling.

As discussed in Section 5 we will differentiate two conditions: in the "full training data" condition, both the out-of-domain and in-domain data are available for training the initial system. In the "growing training data" condition, the initial system is trained only on out-of-domain data.

We will compare the performance of the LHUC method with the "continuation of training" proposed by Freitag and Al-Onaizan (2016). Both methods can start from an already trained system and refine the training on the in-domain data. For the full training data condition we also explore the tagging technique similar to the one proposed by Sennrich et al. (2016).

6.1 Full Training Data Condition

In this data condition, both the WMT and IWSLT training data have been combined together. Each mini-batch in training is selected randomly, so that potentially samples from both domains are presented to the system. In this way there is no implicit domain adaptation effect due to the presentation of the data.

The results, in terms of BLEU score, can be seen in Table 2. Even when the in-domain data is already included in the set used for training the original system, the domain adaptation techniques are able to increase translation quality. Using continuation of training we are able to improve the performance by up to 1.2% absolute. However if we look at the translation quality on the out-of-domain data set we see the catastrophic forgetting effect, with a drop of 1.7 BLEU points.

LHUC achieves an even bigger improvement in translation quality: up to 2.6%. If we would blindly apply the LHUC enhanced system to the out-of-domain data we would again observe the catastrophic forgetting effect, in fact even more pronounced as with continuation of training (number in parenthesis in the corresponding column in Table 2). However in practice this is a non-issue, as the LHUC can easily be deactivated, as discussed in Section 5.

Adding labels to the training data also proves to be an efficient domain adaptation method, with the advantage that it can be combined with the other methods. The improvements due to continuation of training or LHUC are not as big in this case.

For reference, the results of training on in-domain data only have also been included.

6.2 Growing Training Data Condition

In this condition the initial system has only been trained on WMT data. It is also worth noting that this also applies to the BPE vocabulary. It is trained only on the WMT data and then applied to the IWSLT data for the adaptation techniques.

In this case we see even bigger improvements with respect to the baseline system, up to 6 BLEU points. This is mainly due to the baseline system being trained only on out-of-domain data. In this way, in the domain adaptation step, new, unseen training data is added to system. The results are reported in Table 3. Interestingly, the absolute scores of the adapted systems are very close to those reported in Table 2, showing that the domain adaptation techniques can efficiently include new information into the original model. The catastrophic forgetting effect is also more pronounced in this data condition, but we note again that it does not effect the LHUC method.

It is also interesting to note that the performance of the baseline systems on out-of-domain data in both data conditions is the same, indicating that the in-domain data does not really help for the out-of-domain set. This is also indicated by the low score of the in-domain only system on out-of-domain data. Both effects can be explained by comparing the relative sizes of the datasets, as shown in Table 1.

6.3 Efficiency Considerations

As pointed out before, the number of parameters is linear in the number of units in the network. In our specific case we have a total of 2048 units in the encoder (a bidirectional layer with 1024 units in each direction) and 1024 units in the decoder. Using a 32-bit float representation, the overhead of LHUC amounts to just 12KB. For comparison a full model stored on disk in compressed npz format needs 335MB. This shows that LHUC can realistically be used for storing a large amount of

	newstest16 (OOD)	TED 2013 (ID)	TED 2014 (ID)
WMT (OOD) + IWSLT (ID)	33.7	35.9	30.5
+ continuation (ID)	32.0	36.7	31.7
+ LHUC (ID)	33.7 (30.8)	37.9	33.1
OOD + ID labelled	34.4	36.8	32.7
+ continuation (ID)	31.6	37.8	32.9
+ LHUC (ID)	34.4 (32.4)	38.4	33.6
IWSLT only (ID)	16.7	32.4	27.5

Table 2: BLEU scores [%] for the Full Training Data condition. "OOD" denotes out-of-domain data, "ID" denotes in-domain data. For LHUC results, the number in parenthesis shows the result of applying the adapted system to the out-of-domain data (which would not be applied in practice).

	newstest16 (OOD)	TED 2013 (ID)	TED 2014 (ID)
WMT only (OOD)	33.7	31.7	27.3
+ continuation (ID)	30.1	36.8	32.2
+ LHUC (ID)	33.7 (29.7)	37.7	32.8
IWSLT only (ID)	16.7	32.4	27.5

Table 3: BLEU scores [%] for the Growing Training Data condition.

adapted systems.

As for training time, in most experiments the best parameters for LHUC were already achieved in the first checkpoints. On a K80 GPU the best parameter can be found in under one hour for our datasets (continuation needs close to 2h).

7 Conclusions

We have shown how to effectively apply LHUC, a technique first proposed for speaker adaptation in speech recognition, for adapting neural machine translation systems. LHUC achieves good results compared to other domain adaptation methods and due to its low memory footprint and efficient training time can be realistically applied for on-demand adaptation of big systems. In addition it does not suffer the catastrophic forgetting effect, as the LHUC component can be activated or deactivated as needed.

References

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the conference on empirical methods in natural language processing*. Association for Computational Linguistics, pages 355–362.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR* abs/1409.0473.

Nicola Bertoldi and Marcello Federico. 2009. Domain adaptation for statistical machine translation with monolingual resources. In *Proceedings of the fourth workshop on statistical machine translation*. Association for Computational Linguistics, pages 182–189.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, et al. 2016. Findings of the 2016 conference on machine translation. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*. volume 2, pages 131–198.

Marine Carpuat, Hal Daume III, Katharine Henry, Ann Irvine, Jagadeesh Jagarlamudi, and Rachel Rudinger. 2013. Sensespotting: Never let your parallel data tie you to an old domain. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. volume 1, pages 1435–1445.

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, Roldano Cattoni, and Marcello Federico. 2016. The IWSLT 2016 Evaluation Campaign. In

Proceedings of the Internation Workshop on Spoken Language Translation.

Boxing Chen, Colin Cherry, George Foster, and Samuel Larkin. 2017. Cost weighting for neural machine translation domain adaptation. In *Proceedings of the First Workshop on Neural Machine Translation*. pages 40–46.

Chenhui Chu, Raj Dabre, and Sadao Kurohashi. 2017. An empirical comparison of domain adaptation methods for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, pages 385–391. https://doi.org/10.18653/v1/P17-2061.

Markus Freitag and Yaser Al-Onaizan. 2016. Fast domain adaptation for neural machine translation. *arXiv preprint arXiv:1612.06897* .

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. *CoRR* abs/1705.03122. http://arxiv.org/abs/1705.03122.

Felix Hieber, Tobias Domhan, Michael Denkowski, David Vilar, Artem Sokolov, Ann Clifton, and Matt Post. 2017. Sockeye: A Toolkit for Neural Machine Translation. *ArXiv e-prints* .

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. 2016. Google's multilingual neural machine translation system: enabling zero-shot translation. *arXiv preprint arXiv:1611.04558* .

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR* abs/1412.6980.

Michael McCloskey and Neal J Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of learning and motivation* 24:109–165.

Alec Radford, Rafal Jozefowicz, and Ilya Sutskever. 2017. Learning to generate reviews and discovering sentiment. *arXiv preprint arXiv:1704.01444* .

Roger Ratcliff. 1990. Connectionist models of recognition memory: Constraints imposed by learning and forgetting functions. *Psychological review* 97(2):285–308.

Holger Schwenk. 2008. Investigations on large-scale lightly-supervised training for statistical machine translation. In *Proceedings of the Internation Workshop on Spoken Language Translation*. pages 182–189.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Improving neural machine translation models with monolingual data. *arXiv preprint arXiv:1511.06709* .

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Controlling politeness in neural machine translation via side constraints. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, San Diego, California, pages 35–40. http://www.aclweb.org/anthology/N16-1005.

Pawel Swietojanski, Jinyu Li, and Steve Renals. 2016. Learning hidden unit contributions for unsupervised acoustic model adaptation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 24(8):1450–1463.

Pawel Swietojanski and Steve Renals. 2014. Learning hidden unit contributions for unsupervised speaker adaptation of neural network acoustic models. In *Spoken Language Technology Workshop (SLT), 2014 IEEE*. IEEE, pages 171–176.

Marlies van der Wees, Arianna Bisazza, and Christof Monz. 2017. Dynamic data selection for neural machine translation. *CoRR* abs/1708.00712.

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