System description of Supervised and Unsupervised Neural Machine Translation approaches from "NL Processing" team at DeepHack.Babel task

Ilya Gusev ilya.gusev@phystech.edu
MIPT, Dolgoprudny, Moscow Region, 141701, Russian Federation
Artem Oboturov oboturov@gmail.com

Abstract

A comparison [1] of supervised and unsupervised Neural Machine Translation (NMT) models was done for the corpora provided by the DeepHack.Babel competition. It is shown that for even small parallel corpus, fully supervised NMT gives better results than fully unsupervised for the case of constrained domain of the corpus. We have also implemented a fully unsupervised and a semi-supervised NMT models which have not given positive results compared to fully supervised models. A blind set-up is described where participants know at no point what language pair is used for translation, so no extra data could be integrated in pre-submission phase or during training. Finally, future competition organizers should find ways to protect their competition set-ups against various attacks in order to prevent from revealing of language pairs. We have reported two possible types of attacks on the blind set-up.

1 Competition Set-up

The work presented here is motivated by the following observation: an industrial Neural Machine Translation (NMT) system is usually built on a huge parallel corpora and trained for days or even weeks. A "raw" NMT model is then tuned by additional training on client-specific data and by augmentation with some domain-specific information. What if it is not as important to have such a heavy and difficult-to-train model? Instead, why not just use a simple bootstart model based only on the client's data with a subsequent augmentation done using unsupervised learning, which would use any available non-parallel corpora? If such approach would produce results comparable to models trained on large parallel corpora, one could significantly reduce costs of preparing parallel corpora and instead focus on better unsupervised models which work with non-parallel corpora (which are much easier and cheaper to produce). It might also help for the case of low resource languages when no large parallel corpora exist. This paper attempts to answer these questions.

We present the results obtained by the "NL Processing" team in the DeepHack.Babel hackathon [2] on semi-supervised machine translation. Organizers of the competition created a blind set-up - a case in which the source and target languages are not known at any stage of

[1] Images used to train models are publicly available at `https://github.com/aoboturov/loresmt-nlprocessing`

[2] The leaderboard: `http://contest.deephack.me/c/babel/leaderboard`

the competition and the machine translation system should be trained with no specific tuning to the language pair. Language pairs were trained and scored independently, so no one sought to build a universal model. Training and translation were performed by the scoring system. Participants have no insight into the process and could only observe the final score for a submission and/or the failure status. For each language pair participants can submit multiple entries and, based on the scores, adapt their models. Submissions were scored in BLEU-4 (Papineni et al., 2002). The fact that the language pair was not known should have prevented participants from any specific tuning and pre-training of their submitted models; for each submitted model, it had a strict time limit for training and inference (8 hours in total for both stages) and a computational budget constrained by a single dedicated GPU. The participants' models were not allowed to access the internet or any external resources in the training and the evaluation process.

For each language pair the following datasets [3] were available:

- each language of a pair has one monolingual corpora, `1M` sentences;
- a small parallel corpus, `50K` sentence pairs;
- an input corpus to be translated from source to target language, `6K` sentences.

There were 3 language pairs used during the competition: `En-Ru` for test, `Lv-En` for qualification and `En-Ko` for final scoring. Data for training and test are not available publicly and organizers would not release them. Therefore we could only provide a summary [4]: Table 1 describes statistics for the corpora.

Pair	Source Tokens	Source Words	Target Tokens	Target Words
En-Ru	14M	165519	19M	345444
Lv-En	21M	502858	24M	341012
En-Ko	14M	157649	7M	530124

Table 1: Descriptive statistics for the corpora.

The machine translation system could be built as a fully supervised one, though the parallel corpus is small (`50K`); as an unsupervised one, using the two monolingual corpora; and as a semi-supervised one. Given the problem at hand, a simple fully supervised NMT baseline was implemented which was then compared against the Unsupervised Machine Translation Using Monolingual Corpora Only (UNMT) model which was trained both in fully unsupervised and semi-supervised modes.

To prepare for the DeepHack.Babel hackathon we looked into recent supervised NMT systems [5] including: Google's seq2seq (Britz et al., 2017), FAIR Sequence-to-Sequence Toolkit (Gehring et al., 2017), Marian-NMT (Junczys-Dowmunt et al., 2016a) and Sockeye (Hieber et al., 2017). For the competition, however, we focused on the theme of the hackathon, which was on unsupervised and semi-supervised models under the conditions of the blind set-up. The literature review indicated, that the blind set-up itself is novel: (Och et al., 2004), (Tillmann, 2004) and others call their experiments blind with respect to the hold-out set for the final scoring, but we were not able to find an experiment, which was blinded with respect to the language pair.

[3] Contest overview: `http://contest.deephack.me/c/babel/overview`

[4] Samples from the parallel corpora are provided online `https://github.com/aoboturov/loresmt-nlprocessing#the-corpora-extracts`

[5] One could find them on-line: `https://github.com/aoboturov/aoboturov-deephack-babel-qualification`

In Section 2 we outline the baseline that was used to benchmark the UNMT in the blind set-up. In Section 3 we discuss our experiments with the UNMT model for the blind set-up. Finally, in Section 4 we investigate whether prior knowledge of a language pair gives an advantage for the unsupervised NMT approach.

2 Baseline

A supervised NMT model [6] was chosen for the baseline. The model was implemented in OpenNMT (Klein et al., 2017) and had the following Encoder-Decoder architecture:

- the encoder is 3 LSTM layers with a dropout based on 300 dimensional word embeddings for the source language,
- the decoder is stacked LSTM layers with a dropout and a global attention (Luong et al., 2015) based on 300 dimensional word embeddings for the target language.

For each language pair a model was trained only on a `50K` parallel corpus with a `5%` validation set. Data were lowercased and tokenized with Moses (Koehn et al., 2007). Training on an NVIDIA Titan XP GPU usually lasted for 20 to 30 minutes, the results of which are provided in Table 2. Additionally, embeddings were trained with Fasttext (Bojanowski et al., 2017). We have a number of different combinations of LSTM depths and cell-sizes, but we did not search for optimal hyper parameters for the supervised baseline. We have realized that, even without optimal hyperparameters, the baseline beats the UNMT score by an order of magnitude.

On the `Lv-En` language pair, the model performance was mediocre. This could be explained by the fact that `En-Ru` and `En-Ko` were topic-restricted corpora. In particular, both were related to tourism only. On the other hand, the `Lv-En` corpus was extracted from a news feed which had no topic constraints.

3 Unsupervised Neural Machine Translation

The competition included not only parallel corpus for each language pair, but also `1M` monolingual corpus for each language. One way to leverage this data is to use unsupervised NMT model described in Lample et al. (2017). The code for this model is not available, so we built our own implementation [7] based on the PyTorch (Paszke et al., 2017) framework. One can train this model on monolingual corpora using a predefined initial model, which we refer to in this paper as the zero model. The goal of the competition was to find unsupervised and semi-supervised Machine Translation (MT) methods applicable in practice. A fully unsupervised case is covered in Section 3.1, while a semi-supervised approach is described in Section 3.2.

The UNMT [8] would train iteratively using adversarial training (Goodfellow, 2016) with a discriminator [9] presented in Figure 1. In both the semi-supervised and unsupervised cases we ran an unsupervised training epoch which starts from a batch of sentences translated by a model from a previous iteration of unsupervised training (or zero model if it is the first iteration) followed by a noising layer and a pass through the model that has been trained on the current iteration. The preprocessing was done with Moses (Koehn et al., 2007): data were lowercased and tokenized (except for Korean). Figure 2 gives a graphical explanation of the training process.

[6] For a full description of the Encoder-Decoder architecture see `https://github.com/aoboturov/loresmt-nlprocessing#supervised-model-description`.

[7] Implementation of the UNMT: `https://github.com/IlyaGusev/UNMT`

[8] The UNMT model used for translation is described in `https://github.com/aoboturov/loresmt-nlprocessing#unmt-model-description`

[9] The Discriminator description is available online `https://github.com/aoboturov/loresmt-nlprocessing#unmt-discriminator-description`

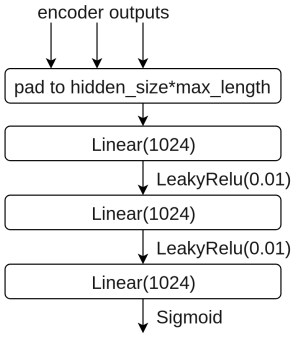

Figure 1: Adversarial training discriminator.

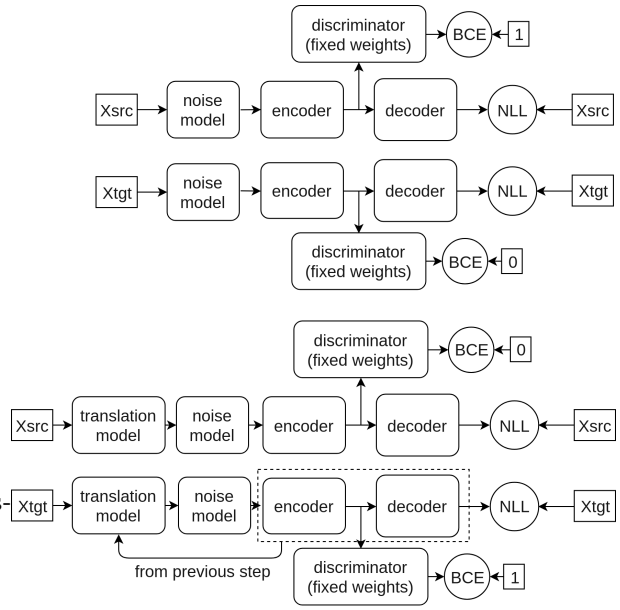

Figure 2: UNMT training process.

There are two types of zero models which we have used: dictionary translation, and a supervised model trained on a small corpora. The translation model has an RNN Encoder-Decoder architecture (Cho et al., 2014) with word embeddings and a global attention (Luong et al., 2015). Figure 3 depicts the encoder and Figure 4 presents the decoder.

3.1 UNMT with a dictionary translation zero model

The dictionary translation model is a translation process which uses a dictionary obtained with an unsupervised embedding (Conneau et al., 2017) (or otherwise an external dictionary could have been used if the language pair was known) to translate each sentence using dictionary translation.

To debug the zero model we first check the input to output copy which is reported in Table 2 as the `In to Out Copy` result. Normally, we would expect an improvement over the `In to Out Copy`, because it is closely related to dictionary translation: words which are not in the dictionary would be copied over from source to target sentences. BLEU scores on language pairs were below 0.01 `BLEU`.

3.2 UNMT with a fully supervised zero model

The fully supervised model was trained in the same way as the baseline. Although the model itself was the Encoder-Decoder from UNMT and not the one from the baseline. The zero model gives a 0.08 `BLUE`, UNMT after adversarial training lasting a day gave results well below 0.01 `BLEU`.

4 Prior Language Pair Information

In this section we describe how the blind set-up can be hacked to improve our results, given that the competition is structured so that the participants do not know the language-pairs being used, and it would be difficult to determine these language pairs within the scope of the competition. The hackathon could have had any pair-combination of 42 languages supported

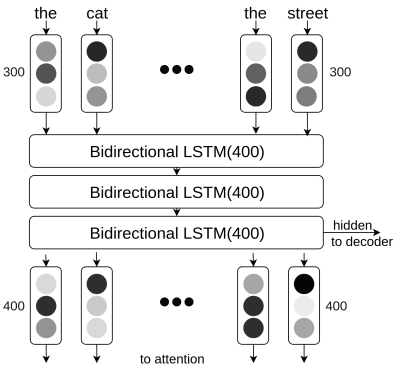

Figure 3: UNMT encoder.

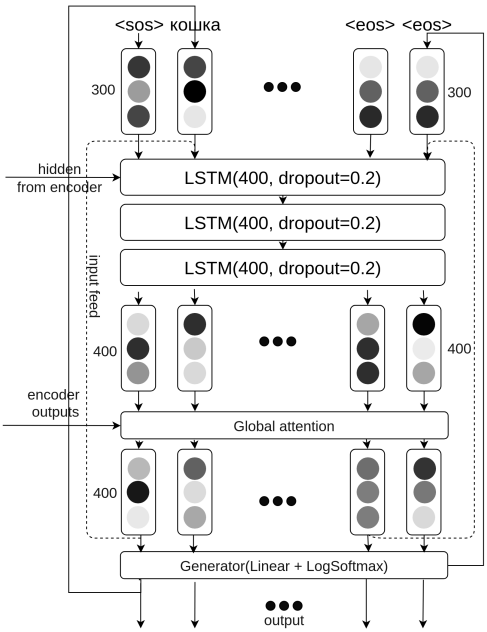

Figure 4: UNMT decoder.

by Booking.com, so the total number of models, if trained unidirectionally, would have been over 1500. Given that even for our simplest baseline model an individual NMT model is at least 300 megabytes, we would have had to train individual unidirectional models, likely for over several days, on some external parallel data, which we did not have for all language pairs, and to package around half of a terabyte of data inside a docker container, which is technologically unrealistic. We could have followed Google's NMT approach (Johnson et al., 2017) or any other MT approach, which have intermediate neural representation, to reduce the total number of models to just one, it should still have to be trained on external parallel corpora, even if they all would be just English to any other of the 41 languages. To reduce the combinatorial complexity of the problem, one could potentially identify the language pair and then just train a single unidirectional model. The competition testing system prevented the access to any external resources and remote calls during training and inference phases. The sheer size of model representations, the total training time, amount and diversity of training data and technical constraints would make pre-training a non-viable option. The only information available to participants was the BLEU score and the failure or success status for the submission. With these information, however, one could devise at least two attacks to identify the language pair and then using this prior knowledge, use it to construct a better translation algorithm.

In Table 2 we reported the best BLEU scores available within the conference submissions for each of the pairs trained on common corpora. On the one hand, we could see that a margin of improvement is just a couple of BLEU points for `En-Ru` and `En-Ko` pairs. On the other hand, `Lv-En` has a very poor result and we would expect that both unsupervised learning and prior knowledge may improve this score.

Below, we describe at least two ways how a language pair identification Side-Channel attack could be executed. The execution time attack is supposed to identify the language pair in a single submission, while the failure status attack would require multiple submissions. The number of submissions used to identify the language pair would matter when the total number

Model	En-Ru Score	Lv-En Score	En-Ko Score
Supervised, 10 Epochs	0.2892	0.0576	0.2542
In to Out Copy	0.0212	0.0208	0.0276
Unsupervised UNMT	-	0.0043	-
Semi-supervised UNMT	-	-	0.0018
Competitors Best	-	0.2334	0.3007
Literature Best, non-blind	0.2980	0.2290	0.2795

Table 2: Evaluation results for models in the blind set-up, measured in BLEU scores.

of submissions for the competition is limited.

4.1 Using Execution Time

There is a way to identify the language pair in one submission by using the side-channel attack technique. In this particular case, the side-channel would be the execution time of the translation algorithm whereby a language identification routine is run on each of the non-parallel corpora and both languages of the pair are detected. Given that the routine could identify N languages, all the pairs could be enumerated to define a mapping of natural numbers in the range $1 \ldots N * (N-1)$. Provided that a specific constant delay is used, one could divide the total execution time by the delay duration to obtain the index of the pair in the mapping.

4.2 Using Failure Status

The second way is a slower combinatorial way in which a failure status is used as an indicator of the language belonging to a subset of languages being tested. A set of all languages identifiable by the routine could be searched in log-time in a breadth-search fashion descending only into subsets where we have established an inclusion relationship.

5 Results and Conclusions

In Table 2, the first four models are the ones that we have produced for the competition, followed by the result reported by the winning team for each round. The last model is reported from literature reviews for the non-blind set-up. The best `Lv-En` and `En-Ru` are from the newstest2017 corpora in Bojar et al. (2017). `En-Ko` is reported from the work of Junczys-Dowmunt et al. (2016b), which uses COPPA corpus. The Literature Best models provide an indicative benchmark for what a MT system trained on a generic parallel corpus might score on a translation task when the language pair is known.

A generic fully unsupervised machine translation problem is hard. In some cases, one could obtain good machine translation models by having a small data set for a limited domain, e.g. for a case of traveling destinations or some other domain-specific translation. Although semi-supervised translation might improve the results, we have not observed that a fully supervised model used as the zero model for the UNMT made any translation improvement over a regular supervised model. For this particular UNMT architecture we report a negative result based on our experiments. Poor performance of the UNMT has to be investigated further, possibly by providing larger non-parallel corpora and changing UNMT model architecture.

References

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Bojar, O., Chatterjee, R., Federmann, C., Graham, Y., Haddow, B., Huang, S., Huck, M., Koehn, P., Liu, Q., Logacheva, V., et al. (2017). Findings of the 2017 conference on machine translation (wmt17). In *Proceedings of the Second Conference on Machine Translation*, pages 169–214.

Britz, D., Goldie, A., Luong, M.-T., and Le, Q. (2017). Massive Exploration of Neural Machine Translation Architectures. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1442–1451.

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). 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.

Conneau, A., Lample, G., Ranzato, M., Denoyer, L., and Jégou, H. (2017). Word translation without parallel data. *arXiv preprint arXiv:1710.04087*.

Gehring, J., Auli, M., Grangier, D., Yarats, D., and Dauphin, Y. N. (2017). Convolutional Sequence to Sequence Learning. In *Proc. of ICML*.

Goodfellow, I. (2016). NIPS 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*.

Hieber, F., Domhan, T., Denkowski, M., Vilar, D., Sokolov, A., Clifton, A., and Post, M. (2017). Sockeye: A Toolkit for Neural Machine Translation. *ArXiv e-prints*.

Johnson, M., Schuster, M., Le, Q. V., Krikun, M., Wu, Y., Chen, Z., Thorat, N., Viégas, F., Wattenberg, M., Corrado, G., et al. (2017). Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Junczys-Dowmunt, M., Dwojak, T., and Hoang, H. (2016a). Is Neural Machine Translation Ready for Deployment? A Case Study on 30 Translation Directions. In *Proceedings of the 9th International Workshop on Spoken Language Translation (IWSLT)*, Seattle, WA.

Junczys-Dowmunt, M., Pouliquen, B., and Mazenc, C. (2016b). COPPA V2. 0: Corpus Of Parallel Patent Applications Building Large Parallel Corpora with GNU Make. In *4 th Workshop on Challenges in the Management of Large Corpora Workshop Programme*, page 15.

Klein, G., Kim, Y., Deng, Y., Senellart, J., and Rush, A. M. (2017). OpenNMT: Open-Source Toolkit for Neural Machine Translation. In *Proc. ACL*.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., et al. (2007). Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 177–180. Association for Computational Linguistics.

Lample, G., Denoyer, L., and Ranzato, M. (2017). Unsupervised machine translation using monolingual corpora only. *CoRR*, abs/1711.00043.

Luong, T., Pham, H., and Manning, C. D. (2015). Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421.

Och, F. J., Gildea, D., Khudanpur, S., Sarkar, A., Yamada, K., Fraser, A., Kumar, S., Shen, L., Smith, D., Eng, K., et al. (2004). A smorgasbord of features for statistical machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in PyTorch.

Tillmann, C. (2004). A unigram orientation model for statistical machine translation. In *Proceedings of HLT-NAACL 2004: Short Papers*, pages 101–104. Association for Computational Linguistics.