PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 111 OCTOBER 2018 125–135

Open Source Toolkit for Speech to Text Translation

Thomas Zenkel, Matthias Sperber, Jan Niehues, Markus Müller,
Ngoc-Quan Pham, Sebastian Stüker, Alex Waibel

Karlsruhe Institute of Technology, Faculty of Informatics, Interactive Systems Labs, Karlsruhe, Germany

Abstract

In this paper we introduce an open source toolkit for speech translation. While there already exists a wide variety of open source tools for the essential tasks of a speech translation system, our goal is to provide an easy to use recipe for the complete pipeline of translating speech. We provide a Docker container with a ready to use pipeline of the following components: a neural speech recognition system, a sentence segmentation system and an attention-based translation system. We provide recipes for training and evaluating models for the task of translating English lectures and TED talks to German. Additionally, we provide pre-trained models for this task. With this toolkit we hope to facilitate the development of speech translation systems and to encourage researchers to improve the overall performance of speech translation systems.

1. Introduction

In recent years a great part of the research interest for automatic speech recognition (ASR) and machine translation (MT) systems focuses on neural approaches. ASR systems used to rely on separate components like a language model, a pronunciation dictionary and an acoustic model and machine translation systems on phrase based approaches. With the emergence of neural networks the implementation of ASR and MT systems became relatively easy, which lead to a number of open-source toolkits for both tasks. Examples for neural ASR toolkits include Eesen (Miao et al., 2015) and Espnet,[1] while XNMT (Neubig et al., 2018) and openNMT-py (Klein et al., 2017) mainly deal with MT systems.

[1] https://github.com/espnet/espnet

 Corresponding author: jan.niehues@kit.edu
Cite as: Thomas Zenkel, Matthias Sperber, Jan Niehues, Markus Müller, Ngoc-Quan Pham, Sebastian Stüker, Alex Waibel. Open Source Toolkit for Speech to Text Translation. The Prague Bulletin of Mathematical Linguistics No. 111, 2018, pp. 125–135. doi: 10.2478/pralin-2018-0011.

Most of these toolkits focus on the evaluation of classical ASR or MT tasks. However, when performing speech translation, the intersection between the different components is an important problem, which rarely receives much attention. ASR systems usually output noisy transcripts without casing and punctuation, which makes the task of the MT system more difficult. End-to-end speech translation systems, that directly translate speech, are mostly difficult to train due to the limited amount of parallel speech translation data.

In this paper we present an open-source toolkit[2] that combines the following components:

- A CTC and an attention based ASR system
- A system to generate the punctuation and the casing of the ASR output
- A neural MT system

We provide recipes to both train and evaluate these models for the task of translating English talks to German. We exclusively use open source data for both training and testing. Pre-trained models of each component are available to download, which makes it possible to replace the individual components without much effort and evaluate the performance directly on speech translation tasks.

2. System Description

The speech translation system presented in this work uses a pipeline approach. According to this pipeline, the audio signal is processed by a sequence of different components to generate the translation. In this work, we use a pipeline of three components. First, the audio is processed by an ASR system to generate a transcript in the source language. In the framework, we integrated two different methods to generate the transcripts. They can be generated by a CTC-based system or by an attentional encoder-decoder based system.

In a second step, punctuation and case information are added to the transcript. This is done by a monolingual translation system based on an attentional encoder-decoder model commonly used in neural machine translation.

The third and last step translates the source text into the target language using a neural machine translation system. We describe the details of all the components in detail in this section.

2.1. Speech Recognition

We include two different speech recognition systems, a CTC based system (Graves et al., 2006) and an attention based system (Bahdanau et al., 2015; Chan et al., 2016). For both approaches we use XNMT (Neubig et al., 2018) to extract 40-dimensional log-Mel-filterbank features with per-speaker mean- and variance-normalization. Both

[2] https://github.com/isl-mt/SLT.KIT

systems transcribe utterances. Our training data is already split into utterances. We rely on the LIUM speaker diarization tool (Rouvier et al., 2013) during testing to create the utterances.

2.1.1. Attentional ASR

Our attentional ASR model follows the listen-attend-spell (Chan et al., 2016) architecture and is similar to the system described by Neubig et al. (2018). The model is implemented with XNMT. Our toolkit exclusively uses XNMT for speech recognition tasks. Compared to a conventional neural machine translation architecture, we replace the encoder with a 4-layer bidirectional pyramidal encoder with a total downsampling factor of 8. The layer size is set to 512, the target embedding size is 64, and the attention uses an MLP of size 128. Input to the model are Mel-filterbank features with 40 coefficients. For regularization, we apply variational dropout of rate 0.3 in all LSTMs, and word dropout of rate 0.1 on the target side (Gal and Ghahramani, 2016). We also fix the target embedding norm to 1 (Nguyen and Chiang, 2018). For training, we use Adam (Kingma and Ba, 2015) with initial learning rate of 0.0003, which is decayed by factor 0.5 if no improved WER is observed. To further facilitate training, label smoothing (Szegedy et al., 2016) is applied. For the search, we use beam size 20 and length normalization with the exponent set to 1.5.

2.1.2. CTC-based ASR

Our CTC-based ASR model is similar to the system described by Zenkel et al. (2018). The input to the model are 40-dimensional Mel-filterbank features. We use every third speech feature of our input sequence and choose the start offset during training randomly, which has the advantage of a lower input sequence length. We train the model to predict Byte Pair Units (Sennrich et al., 2016) [BPE].

The CTC based model consists of four bidirectional LSTM layers with 400 units in each direction followed by a softmax layer. The size of the softmax layer depends on the number of different BPE units we create. We use a dropout rate of 0.25 for all LSTM layers. We train two models based on BPE units with 300 (small model) and 10000 (big model) merges, respectively.

We use SGD with a learning rate of 0.0005 and a momentum term of 0.9 for training. The learning rate is halved whenever the validation token error rate does not decrease by more than 0.1%. We first train the small model and initialize the parameters of the LSTM layers of the big model with the smaller model. We decode the model by greedily selecting the most likely output at each frame.

2.2. Punctuation

Automatic speech recognition (ASR) systems typically do not generate punctuation marks or reliable casing. Using the raw output of these systems as input to MT

causes a performance drop due to mismatched train and test conditions. To create segments and better match typical MT training conditions, we use a monolingual NMT system to add sentence boundaries, insert proper punctuation, and add case where appropriate before translating (Cho et al., 2017).

For both, the punctuation system and the machine translation system, we use the openNMT-py toolkit (Klein et al., 2017) to train the models and to generate the translation. The main difference between the punctuation system and the machine translation system is the input and output data used for training and testing.

The idea of the monolingual machine translation system is to translate from lowercased, unpunctuated text into text with case information and punctuation. Since we do not have any information about the sentence boundaries when inserting the punctuation and case information, we also remove them from the training data. Therefore, in the first step of the pre-processing, we randomly segment the source corpus of the training data into chunks of 20 to 30 words. Based on this randomly segmented corpus, we build the input and output data for the monolingual translation system.

For the input data, we remove all punctuation marks and lowercase all words. Since we will get lower-cased input, we cannot use the same byte-pair encoding (Sennrich et al., 2016) as for the machine translation system. Therefore, we train a separate byte-pair encoding on the lower-cased source data with a code size of 40k. To summarize, the source sequence consists of lower-cased BPE units without any punctuation.

For the target side, we do not want to change the words in the output sentence, but only add case and punctuation information. Therefore, we replace the sentence by features indicating case with punctuation attached. Every word is replaced by a letter *U* or *L*, whether it is upper-cased or lower-cased. Furthermore, punctuation marks following the word are directly attached to the letter.

For example, if the training segment is *I felt worse. Why? I wrote a whole book*, the source input sequence could be *i felt wor@@ se why i wro@@ te a who@@ le book* and the target output sequence will be *U L L. U? U L L L L*.

Based on this method, a translation system is trained to transform the input text into the output tokens. By default, we use the same setup as for the translation system between source and target language.

At test time, we follow the sliding window technique described by Cho et al. (2012). Therefore, we created a test set with segments of length 10 starting with every word on the input data. This means, that except for the beginning and the end of the document, every word occurs ten times, at all positions within the segment. This of course dramatically increases the number of sentences in the test data. In a second step, we generate the target features by applying the monolingual translation system. In a post-processing step, we case the word as it most frequently occurs in the output. We insert punctuation marks, if there is at least one punctuation mark after the word in one of the 10 segments containing this word. If different punctuation marks are predicted, we take the most frequent one. Finally, if the punctuation mark is an end of sentence punctuation mark *{".","!","?"}*, we also start a new segment. The seg-

mented test data with case and punctuation information is passed on to the machine translation system.

2.3. MT

For machine translation, we use a neural machine translation system trained with openNMT-py. By default, we use a rnn-based system.

Within the toolkit, we provide recipes to train a rather small sized translation system. The translation system is trained only on the TED corpus. Due to the limited training data size, we use a smaller model and set the hidden size of the word embeddings as well as for the LSTMs to 512. Furthermore, we use dropout in all the models with a dropout rate of 0.2. In the first training step, we train the model for 10 epochs using Adam optimization. We perform early stopping by evaluating the model after each epoch on the validation data. In the second step, we continue to train the system using a lower learning rate of 0.000125 for another 5 epochs.

The training scripts for the NMT models are provided in *SLT.KIT/scripts/openNMT-py*.

3. Training Data

3.1. ASR

For training the speech recognition systems we use the version 2 of the Tedlium Corpus (Rousseau et al., 2014). We store the log-Mel-filterbank features of each utterances in a hdf5 file.[3] The transcription is stored in a text file one utterance at a time. The key to access the features in the hdf5 file matches the line number of the transcription in the text file. We do not use any additional language modeling data.

3.2. Punctuation and Machine Translation

The machine translation system and the punctuation system are trained on parallel data. We use the TED corpus (Cettolo et al., 2012)[4] and the proceedings from the European Parliament (Koehn, 2005).[5] The source and target sentences are stored in individual text files. As validation data for all systems, we use the dev2010 set provided by IWSLT evaluation campaign (Cettolo et al., 2014).

Prior to translation, we pre-process the data. The default preprocessing includes tokenization, true-casing and byte-pair encoding. The tokenization and true-casing uses the tools from the moses toolkit. The byte-pair encoding is trained on all parallel

[3] https://github.com/h5py/h5py

[4] https://wit3.fbk.eu/

[5] http://statmt.org/europarl/

data and joint codes for both languages are learned. By default, we use a code size of 40k. The scripts to train the true-casing model, to learn the byte-pair encoding and to apply the models to the test data can be found in *SLT.KIT/scripts/defaultPreprocessor*.

4. Evaluation

4.1. Dataset

We evaluate the performance of the ASR and MT systems on English TED talks and its translation to German. We use the test sets used for the IWSLT conference (Cettolo et al., 2014), which are publicly available.[6] We test the input on the following test sets: dev2010, tst2010, tst2013, tst2014.

4.2. Evaluation Metrics

We use Sclite[7] for scoring the ASR output. We calculate the Word Error Rate (WER) based on the provided references in the test sets. Because we do not get a segmentation into utterances, we use talks as the segments for scoring.

In contrast to text translation, the segmentation into sentences is not given a priori in speech translation tasks. Therefore, the standard MT evaluation metrics cannot be applied directly. In this framework, we use the mwerSegmenter[8] to segment the output of the speech translation system according to the reference. In a second step, we can then calculate all machine translation evaluation metrics on the re-segmented output of the translation system.

To evaluate the output, we calculate four different metrics. We generate the BLEU score (Papineni et al., 2002), the TER score (Snover et al., 2006), the BEER metric (Stanojevic and Sima'an, 2014) and CharacTER (Wang et al., 2016). While these metrics are all calculated considering case-information, we also calculate case-insensitive (ci) BLEU and TER scores.

In Table 1 you can find an example sentence processed by our toolkit. First of all the speaker diarization tool generates utterances. These utterances are transcribed by the ASR system. The utterances do not consist of single sentences. The segmentation tool re-segments the output and adds punctuation. In the example this leads to a slightly longer sentence than in the reference, because the expressions “I think” and “we know” are segmented differently. This sentence is then translated to German by the MT system.

[6] https://sites.google.com/site/iwsltevaluation2018/Lectures-task

[7] http://www1.icsi.berkeley.edu/Speech/docs/sctk-1.2/sclite.htm

[8] https://www-i6.informatik.rwth-aachen.de/web/Software/mwerSegmenter.tar.gz

Step	Text
Reference EN	We know that the first 10 years of a career has an exponential impact on how much money you're going to earn.
ASR EN	… don't panic I think this crowd is going to be thought | i think we know that the first ten years of a career has an | exponential impact on how much money you're going turn we know that more than half of | Americans are married …
Punctuation	I think we know that the first ten years of a career has an exponential impact on how much money you're going turn we know.
MT DE	ich denke, wir wissen, dass die ersten 10 Jahre einer Karriere einen entscheidenden Einfluss darauf haben, wie viel Geld wir wissen.
Reference DE	Wir wissen, dass die ersten 10 Jahre eines Berufes eine exponentielle Auswirkung darauf haben, wie viel Geld man verdienen wird.

Table 1. References of an example source and target sentence and the output of the different steps of our system. Utterance boundaries of the ASR system are visualized with the token "|".

4.3. Results

We evaluate the performance of the ASR systems before performing any translation. In Table 2 we report results for the attention based system [Attention], the CTC system with 300 BPE merges [CTC 300] and the CTC system with 10k merges [CTC 10k]. We additionally combine the outputs of the three system by using Rover (Fiscus) [Rover].

Since we do not throw away any segments of the audio file, it still contains segments containing silence. The attention based system tries to produce output for these segments, which leads to a high number of insertion errors. On the other hand, the CTC model handles this situation well and outputs an empty transcript. The CTC 300 model has a higher error rate, which is mostly due to misspelling of words. This can be fixed by training an additional language model as in Zenkel et al. (2018). Combining all three systems improves the results and yields balanced insertion and deletion errors. Many errors of the ASR system are also due to normalization issues between the reference and the hypothesis, especially numbers and dates cause many errors.

Category	S	D	I	WER
Attention	17.5%	2.9%	8.8%	29.2%
CTC 300	17.3%	5.8%	3.6%	26.7%
CTC 10k	13.5%	6.0%	3.3%	22.8%
Rover	13.1%	3.9%	4.2%	21.2%

Table 2. Substitution (S), Insertion (I), Deletion (D) and Word Error Rate (WER) on test2014 for different ASR systems

Category	BLEU	TER	BEER	CharacTER	BLEU(ci)	TER(ci)
Attention	11.88	79.75	41.49	76.98	12.57	77.90
CTC 300	11.49	76.79	40.57	81.52	12.18	75.12
CTC 10k	12.58	74.16	42.05	81.90	13.34	72.36
Rover	13.28	74.34	42.43	78.38	14.01	72.62

Table 3. MT scores on tst2014 for various ASR outputs

We present the results of the MT system in Table 3. Better ASR results also lead to better MT results for all presented metrics. The only exception is the attention based system, which gets better scores than the CTC 300 output when evaluating the performance of the resulting MT output.

Category	dev2010	tst2010	tst2013	tst2014
Attention	13.42	13.57	12.04	11.88
CTC 300	12.33	11.88	12.47	11.49
CTC 10k	13.04	13.44	13.41	12.58
Rover	13.98	14.08	13.73	13.28

Table 4. BLEU scores for different ASR outputs on all test sets

In Table 4 we state the results on the remaining test sets. We do not see a consistent trend if the attention based ASR system or the CTC based system performs better. However, the bigger CTC model consistently yields better BLEU scores than the small CTC model. This can be explained by the higher word error rate and a significantly higher number of miss-spellings in the output of the small model. While the attention based model also yields higher word error rates than the CTC 10k model, this is mostly

due to the higher number of insertion and does not hurt the translation performance as much. We additionally notice that combining the outputs of all ASR systems with Rover improves the results across all test sets.

5. Conclusion

This paper has introduced a toolkit for speech translation. It consistently uses open-source software as well as freely available training and test data. We presented results on test sets for pre-trained models for English to German speech translation. By additionally open-sourcing the trained models we hope to facilitate the task of improving individual components for speech translation systems.

Bibliography

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *ICLR*, 2015. URL https://arxiv.org/pdf/1409.0473v7.pdf.

Cettolo, Mauro, Christian Girardi, and Marcello Federico. WIT^3: Web Inventory of Transcribed and Translated Talks. In *Proceedings of the 16$^{\text{th}}$ Conference of the European Association for Machine Translation (EAMT)*, Trento, Italy, 2012. URL http://hltshare.fbk.eu/EAMT2012/html/Papers/59.pdf.

Cettolo, Mauro, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. Report on the 11th IWSLT evaluation campaign, IWSLT 2014. In *Proceedings of the 11th International Workshop on Spoken Language Translation*, Lake Tahoe, California, 2014. URL http://www.mt-archive.info/10/IWSLT-2014-Cettolo.pdf.

Chan, William, Navdeep Jaitly, Quoc Le, and Oriol Vinyals. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, mar 2016. doi: 10.1109/icassp.2016.7472621. URL https://doi.org/10.1109%2Ficassp.2016.7472621.

Cho, E., J. Niehues, and A. Waibel. Segmentation and punctuation prediction in speech language translation using a monolingual translation system. In *Proceedings of the Ninth International Workshop on Spoken Language Translation (IWSLT)*, 2012. URL https://isl.anthropomatik.kit.edu/pdf/Cho2012.pdf.

Cho, Eunah, Jan Niehues, and Alex Waibel. NMT-Based Segmentation and Punctuation Insertion for Real-Time Spoken Language Translation. In *Interspeech 2017*. ISCA, aug 2017. doi: 10.21437/interspeech.2017-1320. URL https://doi.org/10.21437%2Finterspeech.2017-1320.

Fiscus, J.G. A post-processing system to yield reduced word error rates: Recognizer Output Voting Error Reduction (ROVER). In *1997 IEEE Workshop on Automatic Speech Recognition and Understanding Proceedings*. IEEE. doi: 10.1109/asru.1997.659110. URL https://doi.org/10.1109%2Fasru.1997.659110.

Gal, Yarin and Zoubin Ghahramani. A Theoretically Grounded Application of Dropout in Recurrent Neural Networks. In *Neural Information Processing Systems*

Conference (NIPS), Barcelona, Spain, 2016. URL https://papers.nips.cc/paper/6241-a-theoretically-grounded-application-of-dropout-in-recurrent-neural-networks.pdf.

Graves, Alex, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification. In *Proceedings of the 23rd international conference on Machine learning - ICML '06*. ACM Press, 2006. doi: 10.1145/1143844.1143891. URL https://doi.org/10.1145%2F1143844.1143891.

Kingma, Diederik P. and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations (ICLR)*, Banff, Canada, 2015. URL https://arxiv.org/pdf/1412.6980.pdf.

Klein, Guillaume, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. OpenNMT: Open-Source Toolkit for Neural Machine Translation. In *Proceedings of ACL 2017, System Demonstrations*. Association for Computational Linguistics, 2017. doi: 10.18653/v1/p17-4012. URL https://doi.org/10.18653%2Fv1%2Fp17-4012.

Koehn, Philipp. Europarl: A parallel corpus for statistical machine translation. In *MT summit*, 2005. URL https://homepages.inf.ed.ac.uk/pkoehn/publications/europarl-mtsummit05.pdf.

Miao, Yajie, Mohammad Gowayyed, and Florian Metze. EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, dec 2015. doi: 10.1109/asru.2015.7404790. URL https://doi.org/10.1109%2Fasru.2015.7404790.

Neubig, Graham, Matthias Sperber, Xinyi Wang, Matthieu Felix, Austin Matthews, Sarguna Padmanabhan, Ye Qi, Devendra Singh Sachan, Philip Arthur, Pierre Godard, et al. XNMT: The extensible neural machine translation toolkit. *Conference of the Association for Machine Translation in the Americas (AMTA) Open Source Software Showcase.*, 2018. URL https://arxiv.org/pdf/1803.00188v1.pdf.

Nguyen, Toan and David Chiang. Improving Lexical Choice in Neural Machine Translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Association for Computational Linguistics, 2018. doi: 10.18653/v1/n18-1031. URL https://doi.org/10.18653%2Fv1%2Fn18-1031.

Papineni, Kishore, 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*. Association for Computational Linguistics, 2002. doi: 10.3115/1073083.1073135. URL https://www.aclweb.org/anthology/P02-1040.pdf.

Rousseau, Anthony, Paul Deléglise, and Yannick Esteve. Enhancing the TED-LIUM Corpus with Selected Data for Language Modeling and More TED Talks. In *LREC*, 2014. URL http://www.lrec-conf.org/proceedings/lrec2014/pdf/1104_Paper.pdf.

Rouvier, Mickael, Grégor Dupuy, Paul Gay, Elie Khoury, Teva Merlin, and Sylvain Meignier. An open-source state-of-the-art toolbox for broadcast news diarization. In *Interspeech*, 2013. URL http://publications.idiap.ch/downloads/reports/2013/Rouvier_Idiap-RR-33-2013.pdf.

Sennrich, Rico, Barry Haddow, and Alexandra Birch. 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)*. Association for Computational Linguistics, 2016. doi: 10.18653/v1/p16-1162. URL https://doi.org/10.18653%2Fv1%2Fp16-1162.

Snover, Matthew, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas*, 2006. URL https://www.cs.umd.edu/~snover/pub/amta06/ter_amta.pdf.

Stanojevic, Milos and Khalil Sima'an. BEER: BEtter Evaluation as Ranking. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*. Association for Computational Linguistics, 2014. doi: 10.3115/v1/w14-3354. URL https://doi.org/10.3115%2Fv1%2Fw14-3354.

Szegedy, Christian, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, jun 2016. doi: 10.1109/cvpr.2016.308. URL https://doi.org/10.1109%2Fcvpr.2016.308.

Wang, Weiyue, Jan-Thorsten Peter, Hendrik Rosendahl, and Hermann Ney. CharacTer: Translation Edit Rate on Character Level. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*. Association for Computational Linguistics, 2016. doi: 10.18653/v1/w16-2342. URL https://doi.org/10.18653%2Fv1%2Fw16-2342.

Zenkel, Thomas, Ramon Sanabria, Florian Metze, and Alex Waibel. Subword and Crossword Units for CTC Acoustic Models. *Interspeech*, 2018. URL https://arxiv.org/pdf/1712.06855.pdf.

Address for correspondence:
Jan Niehues
jan.niehues@kit.edu
Karlsruhe Institute of Technology
Adenauerring 2,
76131 Karlsruhe, Germany