This document is part of the Research and Innovation Action "European Live Translator (ELITR)".
This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No 825460.

Deliverable D2.1

Report 1 on Initial ASR Systems

Jonáš Kratochvíl (CUNI), Thai-Son Nguyen (KIT), Peter Polák (CUNI), Sebastian Stüker (KIT)

Dissemination Level: Public

Version 1.0, 30th June, 2020

Grant agreement no.	825460
Project acronym	ELITR
Project full title	European Live Translator
Type of action	Research and Innovation Action
Coordinator	Doc. RNDr. Ondřej Bojar, PhD. (CUNI)
Start date, duration	1st January, 2019, 36 months
Dissemination level	Public
Contractual date of delivery	30th June, 2020
Actual date of delivery	30th June, 2020
Deliverable number	D2.1
Deliverable title	Report 1 on Initial ASR Systems
Type	Report
Status and version	Version 1.0
Number of pages	20
Contributing partners	KIT, CUNI
WP leader	KIT
Author(s)	Jonáš Kratochvíl (CUNI), Thai-Son Nguyen (KIT), Peter Polák (CUNI), Sebastian Stüker (KIT)
EC project officer	Alexandru Ceausu
The partners in ELITR are:	• Univerzita Karlova (CUNI), Czech Republic • University of Edinburgh (UEDIN), United Kingdom • Karlsruher Institut für Technologie (KIT), Germany • PerVoice SPA (PV), Italy • alfatraining Bildungszentrum GmbH (AV), Germany
Partially-participating party	• Nejvyšší kontrolní úřad (SAO), Czech Republic

For copies of reports, updates on project activities and other ELITR-related information, contact:

Doc. RNDr. Ondřej Bojar, PhD., ÚFAL MFF UK — bojar@ufal.mff.cuni.cz
Malostranské náměstí 25 — Phone: +420 951 554 276
118 00 Praha, Czech Republic — Fax: +420 257 223 293

Copies of reports and other material can also be accessed via the project's homepage:

http://www.elitr.eu/

Contents

1 Executive Summary

In this deliverable we describe the initial automatic speech recognition (ASR) systems that we created for the ELITR speech translation system. We have investigated different types of models: hybrid HMM/ANN models, encoder-decoder with attention based sequence-to-sequence (S2S) and self-attention based S2S models. We have created the models in such a way as to meet the primary design criteria of the ASR systems of the ELITR speech translation system: stream processing, real-time processing and low-latency processing. For the inital systems we have produced traditional HMM/ANN recognition systems, and a proof-of-concept S2S model for English that advances the state-of-the-art.

We made the initial ASR systems available for the languages Czech, English, German, French, Spanish and Italian.

As a next step we will advance the S2S workers to a level of maturity that they can be deployed in the ELITR speech translation system.

2 Introduction

This deliverable presents the initial automatic speech recognition (ASR) systems developed for ELITR's speech translation system. In order for the ASR systems to work in a machine interpretation set-up, they need to fulfill the following criteria:

Streaming Mode: The systems have to work on streaming ASR input. That means that the ASR data is not processed in batch mode; instead a continuous stream of audio needs to be continuously recognized.

Real-time or faster: The systems have to process the incoming audio in real-time or faster in order to be able to keep up with the incoming speech.

Low-latency: The systems have to output their results in as low a latency as possible, in order for the audience to be able to follow the speeches in sync with the speakers, their presentations, and the results of the automatic interpretation.

The creation of the initial ASR systems for ELITR came in a period of dramatic changes when it comes to ASR systems in general, and low-latency, real-time ASR systems specifically.

While at the on-set of the project the state-of-the-art consisted of HMM based speech recognition systems that use feed-forward neural networks for the estimation of the HMMs' emission probabilities, during the first year of the project advances in sequence-to-sequence (seq2seq) speech recognition technology led to better performance of these models in offline settings. Therefore, we started to create seq2seq systems that not only perform better in offline settings, but also fulfill the specifications above in order for the systems to work in the ELITR setting.

While, in general, leading to drastically lower word error rates, seq2seq systems also have some disadvantages:

Data needs: Current seq2seq systems tend to need more data in order to surpass HMM based systems. i.e., when trained on large amounts of data seq2seq systems clearly outperform HMM systems, but when trained on smaller amounts of data, HMM systems might be able to outperform seq2seq systems.

End-to-end data Seq2seq systems are exclusively trained on transcribed audio data. For HMM based systems this is only the case for the acoustic model, while the language model is trained on text only. This is an advantage because a) text data is often available in larger amounts than transcribed audio data, and b) domain specific text is often more readily available than domain specific transcribed audio data.

Vocabulary and language model adaptation Our seq2seq models are open vocabulary models. Therefore, manually adapting the vocabulary to account for domain specific terms is not easily doable (at the moment). Also, as the seq2seq models do not have an explicit language model, adapting the systems on text only is not possible (at the moment).

3 Types of systems investigated

3.1 Hybrid HMM/ANN ASR with the JRTk

The conventional approach to statistical automatic speech recognition (ASR) uses a Bayes classifier and Hidden Markov Model (HMM) to decouple the modeling posterior $P(W \mid X)$, i.e. the probability of a textual transcript W given acoustic feature X, into two independent probabilities $P(X \mid W)$ and $P(W)$. Then, these probabilities are modeled separately by an acoustic model and a language model in training and combined within a Viterbi beam search for inference. The recent advances of artificial neural networks (ANN) in both acoustic modeling and language modeling have made the hybrid HMM/ANN approach dominant in many types of ASR applications.

We have used the Janus Recognition Toolkit (JRTk) (Finke et al., 1997) for the development of HMM/ANN ASR systems for the initial phase of the ELITR project. JRTk includes the IBIS dynamic decoder (Soltau et al., 2001), which allows inference to run in real-time which is the key component to build low-latency and streaming ASR systems. The adoption of a HMM/ANN ASR is very useful for speech translation tasks from low-resource languages as the recognition accuracy can be improved via the adaptation of both the acoustic and language models on small amounts of in-domain data.

For each supported language (see Section 4), we collected a set of speech data with available transcripts for training the acoustic model and text data for training the language model. We then built individual ASR systems for different input languages in the ELITR multilingual speech translation system. In the setup, the acoustic models are implemented using feed forward neural networks (FFNN) with 6 layers of 1600 units (or 1024 units for the languages with smaller amounts of training data) so as to meet the performance and speed requirements. For language modeling, we trained and used 4-gram language models in all the systems.

3.2 End-to-end Sequence-to-sequence ASR

Attention-based sequence-to-sequence (S2S) models, which use a neural network architecture to approximate the direct mapping from the acoustic signal to the textual transcript, have become a very efficient approach for building high performance speech recognition systems, as in batch processing on GPUs they have a very low real-time factor while at the same time having a significantly lower word error rate Chan et al. (2016); Bahdanau et al. (2016). The advantage of the S2S approach lies in its simplification of training an entire speech recognition system, thereby hiding the awareness of complicated components as in the HMM-based systems. A typical S2S network architecture has three basic components:

Encoder The encoder is analogous to a conventional acoustic model, which takes the input features and maps them to a higher-level feature representation.

Decoder Given the encoder's output the decoder estimates a distribution over output tokens.

Attention The time-alignment between the encoder and the decoder is handled by the attention mechanism.

We have investigated how to build such an S2S system for the ASR part of a speech translation system. As S2S models require typically a large amount of training data for their efficiency, we started with publicly available speech data sets and focused on English. The initial research has been published in Nguyen et al. (2020b) in which we have proposed a setup for building high-performance S2S systems with two different architectures: a) an LSTM network in both the encoder and the decoder b) a model that uses self-attention in all components. In the same study, we have shown that on the Switchboard and Fisher telephone conversation benchmarks, our proposed S2S ASR systems outperform the best reported HMM/ANN ASR in the offline condition in two training settings: 300 and 2000 hours of speech. We also provided a different S2S ASR system trained on a non-native speech dataset to the ELITR submission on the non-native speech translation task of IWSLT 2020 Ansari et al. (2020). The results from the submission have potentially shown that the S2S systems perform very well in recognizing non-native speech.

While the S2S models have been shown to outperform other approaches on standard speech recognition tasks in an offline setup, they face several challenges when having to work in online mode (i.e., when the complete audio data is not available before processing). For the deployment of the proposed S2S ASR model in a run-on and streaming setting our analysis in Nguyen et al. (2020a) shows that the attention mechanism, the encoder component and the standard beam-search inference encounter latency issues . To mitigate these problems, we introduced an additional loss function controlling the uncertainty of the attention mechanism, a modified beam search identifying partial, stable hypotheses, ways of working with BLSTMs in the encoder, and the use of chunked BLSTMs. In the experiments with telephone communication speech, we showed that with a delay of 1.5 seconds in all output elements, our proposed streaming recognizer can achieve the ideal performance of an offline system with the same configuration.

3.3 Enhanced ASR with phoneme-level intermediate step

Another requirement on the ASR is to provide robust performance in various situations. Besides of the bad acoustics or low-quality hardware, the pronunciation of a speaker can have significant impact on resulting quality of the transcripts. We therefore started to investigate another approach targeted at improving robustness to phonetic variations resulting from accents and dialects.

Motivated by recent work by Salesky et al. (2019) and Hrinchuk et al. (2019), we explored a robust ASR pipeline consisting of two components — an acoustic model recognizing phonemes, and a phoneme-to-grapheme translation model. We decided to use phonemes as the intermediate representation between the acoustic and the translation model because we believe that the conventional grapheme representation is too constrained with complicated rules of mapping speech to a transcript. This issue becomes immense when dealing with dialects and non-native speakers. In such case, the ASR may guess grapheme (standard) word that is "far" from the pronounced word. The acoustic model of the proposed system should output phoneme transcript matching pronunciation and the following phoneme-to-grapheme model should output a corrected transcript based on the context.

First, we investigated how well the proposed pipeline performs when trained on clean data. Notably, we trained the phoneme-to-grapheme translation model on a non-speech text corpus artificially translated to phonemes. The trained models did not outperform the baseline. But to our surprise, its performance is only slightly worse.

To support robustness, we further investigated several training and fine-tuning schemes using transfer from related task (SLT) and "corrupted" data. We gathered the "corrupted" acoustic data by inferring speech corpora using an ensemble of acoustic models for each language (English and Czech). The obtained "corrupted" transcripts where then paired with golden transcripts and used in the training of the translation model.

Further, we initialized phoneme-to-grapheme translation model with weights from SLT task (i.e., the encoder with weights from Cs-to-En SLT and the decoder with En-to-Cs SLT for Czech ASR, and vice versa for English ASR).

We made two key observations:

1. using "corrupted" data for training helps to reduce the WER,
2. transfer learning from SLT further promotes the robustness of the ASR.

Further details can be found in Polák (2020). We also participated with this system (see Polák et al. (2020), reproduced here in Appendix A) in the Non-Native Speech Translation Task for IWSLT 2020 (Ansari et al., 2020).

3.4 Domain adaptation of Czech speech recognition

One of the challenges of automatic speech recognition is adaptation of the system trained on a large amount of general acoustic data to a specific target domain. As part of our research, we have developed a scalable domain adaptation pipeline for Czech, aiming to reduce the gap in performance between various domains. Our adaptation techniques mainly focus on the language model and lexicon adaptation. We conduct experiments described in this section with the Kaldi toolkit Povey et al. (2011).

Test sets To test our adaptation techniques, we have prepared five distinct test sets from various domains. These include Czech parliament hearings (PS), Euro parliament debates (EP), presentations about computational linguistics (CS), Conference of Supreme Audit Office in Prague (SAO), and talks from Czech broadcast station Český Rozhlas (ČRO). We include details of our test sets in Table 1.

Test set	EP	CL	ČRO	PS	SAO
Total hours	2:45:25	1:46:01	2:06:41	2:44:58	0:41:08
Total words	17051	12644	18363	20592	3759
Unique words	4381	2746	4362	3847	1375
Unique speakers	4	1	8	53	1
Total distinct talks	3	2	6	1	1

Table 1: Test set statistics

Domain adaptation algorithm We first identify the target domain and extract as many existing text materials as possible. These typically include slides of a particular presentation, abstract of a relevant paper or news articles, and keywords relevant to the target domain. Using the obtained set of sentences, we embed them into a latent space using the sentence embedding technique described in Arora et al. (2016). As our main corpus, we use SumeCzech (Straka et al., 2018). This corpus comprises approximately one million Czech articles collected from various online news sites such as “Novinky.cz”, “Denik.cz” or “Idnes.cz”. Each article is divided into *title*, *abstract* and *full-text*. We divide each abstract and title into sentences, and for each of them remember the mapping to the full-text article to which they belong. We then embed these sentences obtained from abstracts and titles using the same sentence embedding algorithm. For each of our collected domain sentence, we compute cosine similarity with all the abstracts and title sentences from SumeCzech and select n (this is a hyper parameter of our algorithm that can be specified by the user) most similar ones. Finally, we map these most similar titles and abstract sentences back to their full-text articles.

The second part of our algorithm uses two existing applications developed by the Czech National Corpus. These are KonText[1] and KWords[2]. We take the text collection obtained from the first part of the algorithm and search for domain-specific words inside it by KWords application. It compares the relative frequencies of each word in the input text against frequencies in the selected backend corpus. We then compute the χ^2 and log-likelihood statistical tests on the frequency differences, and we sort the words in descending order from the most domain-specific to the least domain-specific. This way, we extract the m (this is again hyperparameter of our algorithm) most significant words from our domain text corpus. We then query each of these words into the KonText application. It searches a large text corpus and outputs all available contexts for a particular word. We extract k (this parameter is usually set to 100-200) context sentences for each query word and include all collected contexts into our domain texts from the first part of the algorithm.

Finally, we train a domain-specific n-gram language model and interpolate it with a larger and

[1] `https://kontext.korpus.cz/`
[2] `https://kwords.korpus.cz/`

more general n-gram model. The domain words also extend the model lexicon, and we recompile the Kaldi decoding graph using this new interpolated language model and its lexicon.

Results In Table 2 we present the results of all evaluated systems on all our test sets. We compare our models against the Google Cloud Czech ASR[3] and against the UWebASR (Švec et al., 2018) provided by the University of West Bohemia.[4] As our systems, we first present the baseline model, which uses the general language model trained on three million news sentences. We adapt the *N-gram adaptation* model by the algorithm described in the previous section to the domain of each of our test sets (resulting in 5 distinct models). We also extend the model lexicon by the domain words. Lastly, we extend the *N-gram adaptation* model by recurrent neural network lattice re-scoring. The architecture of the recurrent language model is described in Xu et al. (2018). The RNN language model was also trained using the interpolation of the domain-specific and general text data.

Model type	**EP**	**CL**	**ČRO**	**PS**	**SAO**
Google Cloud	9.562	6.371	14.729	9.646	3.383
UWB	14.133	7.447	9.425	8.556	4.334
Baseline	6.606	3.933	6.737	9.771	3.591
N-gram adaptation	6.780	3.140	6.587	9.389	1.519
RNNLM adaptation	5.972	3.138	6.475	9.389	1.493

Table 2: Word error rates of all considered systems and test sets.

From Table 2 we see that the domain adaptation can help improve model performance across a diverse set of domains. In the future, we plan to experiment with acoustic adaptation, where we could adapt the neural network acoustic model to the voice of a specific speaker or to a particular acoustic condition.

[3]`https://cloud.google.com/speech-to-text`
[4]https://lindat.mff.cuni.cz/services/uwebasr/

4 Language Coverage

Using the technologies above we were able to cover the languages: Czech, English, German, French, Spanish and Italian.

Table 3 gives an overview for which languages which types of systems are available at this point and the total amount of training data used for creating the workers. The training data used is generally a mixture of data from the European Parliament and Broadcast News. For English, additional data from the Technology, Entertainment, Design (TED) conferences and from telephone conversations (Switchboard and Fisher data) was available.

Language	**# Training Data**	**Types of Workers**
Czech	444	HMM/ANN
English	1500	HMM/ANN, enc/dec
German	450	HMM/ANN
French	268	HMM/ANN
Spanish	272	HMM/ANN
Italian	178	HMM/ANN

Table 3: Languages serviced by ASR workers, the amount of training data they were trained on in hours and the type of model used: hybrid HMM/ANN (HMM/ANN) model, S2S encode-decoder with attention model (enc/dec)

5 Conclusion

In this deliverable we have described the initial ASR system developed for the ELITR speech translation system. We have investigated a variety of different modeling techniques, such as hybrid HMM/ANN systems, encoder-decoder with attention based sequence-to-sequence models and self-attention models. We also investigated domain adaptation techniques for hybrid models and tested them on Czech.

Using these different models, we were able to cover the languages English, German, French, Spanish, and Italian and provide suitable HMM/ANN workers for the ELITR speech translation system. The next step will now be to continue the work on sequence-to-sequence models for which we have shown that we can deploy the in streaming mode as required for ELTIR, and to produce workers that are mature enough to be deployed in the ELITR framework.

References

Ebrahim Ansari, Amittai Axelrod, Nguyen Bach, Ondrej Bojar, Roldano Cattoni, Fahim Dalvi, Nadir Durrani, Marcello Federico, Christian Federmann, Jiatao Gu, Fei Huang, Kevin Knight, Xutai Ma, Ajay Nagesh, Matteo Negri, Jan Niehues, Juan Pino, Elizabeth Salesky, Xing Shi, Sebastian Stüker, Marco Turchi, and Changhan Wang. Findings of the IWSLT 2020 Evaluation Campaign. In *Proceedings of the 17th International Conference on Spoken Language Translation (IWSLT 2020)*, Seattle, USA, 2020.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. A simple but tough-to-beat baseline for sentence embeddings. 2016.

D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio. End-to-end attention-based large vocabulary speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4945–4949, 2016.

W. Chan, N. Jaitly, Q. Le, and O. 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)*, pages 4960–4964, 2016.

Michael Finke, Petra Geutner, Hermann Hild, Thomas Kemp, Klaus Ri es, and Martin Westphal. The karlsruhe VERBMOBIL speech recognition engine. In *Proc. of ICASSP*, 1997.

Oleksii Hrinchuk, Mariya Popova, and Boris Ginsburg. Correction of automatic speech recognition with transformer sequence-to-sequence model. *arXiv preprint arXiv:1910.10697*, 2019.

Thai-Son Nguyen, Ngoc-Quan Pham, Sebastian Stueker, and Alex Waibel. High performance sequence-to-sequence model for streaming speech recognition. *arXiv preprint arXiv:2003.10022*, 2020a.

Thai-Son Nguyen, Sebastian Stüker, Jan Niehues, and Alex Waibel. Improving sequence-to-sequence speech recognition training with on-the-fly data augmentation. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7689–7693. IEEE, 2020b.

Peter Polák. Spoken language translation via phoneme representation of the source language. Master's thesis, Charles University, 2020.

Peter Polák, Sangeet Sagar, Dominik Macháček, and Ondřej Bojar. Neural ASR with Phoneme-Level Intermediate Step – A Non-Native Speech Translation Task Submission to IWSLT 2020. In *Proceedings of the 17th International Conference on Spoken Language Translation (IWSLT 2020)*, Seattle, USA, 2020.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, and Karel Vesely. The kaldi speech recognition toolkit. In *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, December 2011. IEEE Catalog No.: CFP11SRW-USB.

Elizabeth Salesky, Matthias Sperber, and Alan W Black. Exploring phoneme-level speech representations for end-to-end speech translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1835–1841, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1179. URL `https://www.aclweb.org/anthology/P19-1179`.

Hagen Soltau, Florian Metze, Christian Fügen, and Alex Waibel. A one-pass decoder based on polymorphic linguistic context assignment. In *Automatic Speech Recognition and Understanding, 2001. ASRU'01. IEEE Workshop on*, pages 214–217. IEEE, 2001.

Straka, Mediankin, Kocmi, Žabokrtský, Hudeček, and Hajič. Sumeczech, 2018.

Xu, Li, Wang, Wang, Kang, Chen, Povey, and Khudanpur. Neural network language modeling with letter-based features and importance sampling. pages 6109–6113. IEEE, 2018.

Švec, Bulín, Pražák, and Ircing. Uwebasr - web-based asr engine for czech and slovak, 2018.

A CUNI Neural ASR with Phoneme-Level Intermediate Step for Non-Native SLT at IWSLT 2020

CUNI Neural ASR with Phoneme-Level Intermediate Step for Non-Native SLT at IWSLT 2020

Peter Polák and **Sangeet Sagar** and **Dominik Macháček** and **Ondřej Bojar**
Charles University
Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics
{polak,sagar,machacek,bojar}@ufal.mff.cuni.cz

Abstract

In this paper, we present our submission to the Non-Native Speech Translation Task for IWSLT 2020. Our main contribution is a proposed speech recognition pipeline that consists of an acoustic model and a phoneme-to-grapheme model. As an intermediate representation, we utilize phonemes. We demonstrate that the proposed pipeline surpasses commercially used automatic speech recognition (ASR) and submit it into the ASR track. We complement this ASR with off-the-shelf MT systems to take part also in the speech translation track.

1 Introduction

This paper describes our submission to Non-Native Speech Translation Task in IWSLT 2020 (Ansari et al., 2020). We participate in two sub-tracks: offline speech recognition and offline speech translation from English into Czech and German.

We focus on the speech recognition, proposing a robust pipeline consisting of two components — an acoustic model recognizing phonemes, and a phoneme-to-grapheme translation model, see Figure 1. We decided to use phonemes as the intermediate representation between the acoustic and the translation model because we believe that conventional grapheme representation is too constrained with complicated rules of mapping speech to a transcript. This issue becomes immense when dealing with dialects and non-native speakers.

Both models used in our pipeline are end-to-end deep neural networks, Jasper (Li et al., 2019) for the acoustic model and Transformer (Vaswani et al., 2017) for the phoneme-to-grapheme translation model.

For punctuating, truecasing, segmenting and translation into Czech and German, we use off-the-shelf systems provided by ELITR project.

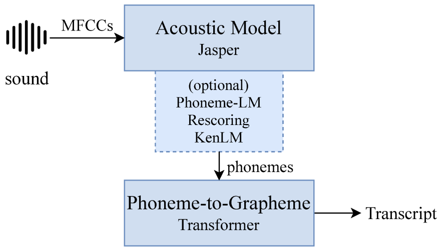

Figure 1: The architecture of proposed model.

The paper is organized as follows: Section 2 reviews related work. In Sections 3 and 4 we describe models for our speech recognition pipeline and their training. In Section 5, we describe the punctuator, truecasor and segmenter, and machine translation into Czech and German in Section 6. We summarize our submissions in Section 7 and conclude in Section 8.

2 Related Work

This section reviews the related work.

2.1 Phonemes and Acoustic Models

Phones and phonemes are well-established modelling units in ASR. They have been used since the beginning of the technology in 1950s (Juang and Rabiner, 2005), for an empirical comparison of different linguistic units for sound representation, see Riley and Ljolje (1992).

An important work popularizing neural networks in ASR to phonemes is Waibel et al. (1989). This work proposes using a time-delayed neural network (TDNN) to model acoustic-phonetic features and the temporal relationship between them. The authors demonstrate that the proposed TDNN can learn shift-invariant internal abstraction of speech and use it to make a robust final decision.

191

Proceedings of the 17th International Conference on Spoken Language Translation (IWSLT), pages 191–199
July 9-10, 2020.
https://doi.org/10.18653/v1/P17

Salesky et al. (2019) suggest using of phoneme-based ASR in speech translation. Their end-to-end speech translation pipeline first obtains phoneme alignment using the deep neural network hidden Markov models (DNN-HMM) system and then averages feature vectors with the same phoneme for consecutive frames. Phonemes outputted by DNN-HMM then serve as input features for speech translation.

2.2 Phoneme-to-Grapheme Models

In most past studies that included a separate phoneme-to-grapheme (P2G) translation component into the ASR, the phoneme representation was used only for out-of-vocabulary (OOV) words, see, e.g. Decadt et al. (2001); Horndasch et al. (2006); Basson and Davel (2013).

Decadt et al. (2001) apply phoneme-to-grapheme to enhance the readability of OOV output in Dutch speech recognition. In their setup, the ASR outputs standard (orthographic) text for known words. For OOVs, phonemes are outputted. Because the phonemes are unreadable for most users, the authors translate phonemes using memory-based learning. The word error rate of this improved setup of Dutch ASR was actually higher than the baseline, on the other hand, the output was better readable for an untrained person. They report that 41 % of words were transcribed with at most one error, and 62 % have only two errors. Furthermore, most of the incorrectly transcribed words do not exist in Dutch.

Horndasch et al. (2006) introduce a data-driven approach called MASSIVE. Their main objective is to find appropriate orthographic representations for dictated Internet search queries. Their system iteratively refines sequences of symbol pairs in different alphabets. In the first step, they find the best phoneme-grapheme alignment using the expectation-maximization algorithm. In the second step, they cluster neighbouring symbols together to account for insertions. Finally, n-gram probabilities of symbol pairs are learned. During the inference, the input string is split into individual symbols. All possible symbol pairs are generated for each symbol, and the best sequences are selected in a beam search.

2.3 Error Correction in ASR

Hrinchuk et al. (2019) deal with the correction of errors in ASR by introducing Transformer post-processing. The authors first train an ensemble of 10 ASR models. Using these models, they collect "ASR corrupted" data. Subsequently, they train a Transformer on this data where the "ASR corrupted" text serves as the source and the original true transcripts as the target. In their best setup, they utilize transfer learning. They use BERT (Devlin et al., 2018), a masked language model consisting only of Transformer encoder, and initialize both encoder and decoder of their Transformer correction model with BERT's weights.

2.4 Online ASR Services

We compare our work with Google Cloud Speech-to-Text API[1] and Microsoft Azure Speech to Text.[2] Both of these services provide publicly available APIs for transcribing audio recordings.

3 Neural ASR with Phoneme-Level Intermediate Step

Our main idea is to couple an end-to-end acoustic model with a specialized "translation" model, which translates phonemes to graphemes and corrects the ASR errors.

The motivation for the translation step is that the translation model can exploit larger context than a basic convolutional acoustic model. Furthermore, we can utilize considerably larger non-speech corpora to train this part of the pipeline.

3.1 Acoustic Model

For our acoustic model, we use the Jasper (Li et al., 2019) convolutional neural architecture in the variant of Jasper DR 10x5 variant, as described in the original paper. It is implemented within the NeMo library (Kuchaiev et al., 2019).

For training, we use approximatelly 1 000 hours of speech data from LibriSpeech (Panayotov et al., 2015) and 1 000 hours of Common Voice[3]. Because we want the model to produce phonemes and not graphemes, which are available in the training corpora, we converted the transcript to IPA phonemes using the `phonemizer`[4] tool.

To speed-up the training process, we initialize our English sound-to-phoneme Jasper model with

[1] https://cloud.google.com/speech-to-text
[2] https://azure.microsoft.com/en-us/services/cognitive-services/speech-to-text/
[3] https://voice.mozilla.org/en
[4] https://github.com/bootphon/phonemizer

Type	Corpus	Adapt.	Full training
dev	LS Clean	46.07	3.84
	CV	54.69	11.86
test	LS Clean	-	4.18 / 4.48† / 3.58‡
	LS Other	-	11.48 / 11.67† / 8.57‡
	CV	-	10.21 / 10.47† / 6.46‡

Table 1: Results in % of *Phoneme* Word Error Rate (PWER) using greedy decoding (no mark), beam search (†) and beam search with language model (‡). The language model is trained on phonemized ASR training data. Note, PWER is not directly comparable to WER. "LS" LibriSpeech. "CV" Common Voice.

the available checkpoint of the standard sound-to-grapheme model.[5]. This seed model was trained on LibriSpeech, Mozilla Common Voice, WSJ, Fisher, and Switchboard corpora, which is beyond the set of corpora allowed for a constrained submission. The model yields word error rate (WER) of 3.69 % on LibriSpeech test-clean, and 10.49 % on test-other using greedy decoding.

For a smooth transition from the Latin alphabet to IPA, we start our training with an adaptation phase of 2,000 training steps. As the model's memory footprint is smaller during this phase, we increase the batch size to 64 (global batch size is 640). One thousand steps are warm-up; the maximal learning rate is 0.004.

The full training takes ten epochs. The model memory requirements increase, therefore we reduce the batch size to 16 (global batch size is 160). We also reduce the learning rate to 0.001.

Optionally, we include a phoneme-level language model, which re-scores the output of the acoustic model before the phoneme-to-grapheme translation, to achieve higher quality. Setups that use this component are further in this paper marked with "_lm".

Results of training after the Adaptation phase (the "Adaptation" column) and the Full training are in Table 1. Note that these scores are calculated on the reference transcript converted to phonemes using phonemizer. Token ambiguities thus change, and these scores are not comparable to standard grapheme WER.

The training is executed on 10 NVIDIA GTX 1080 Ti GPUs with 11 GB VRAM.

[5] https://ngc.nvidia.com/catalog/models/nvidia:multidataset_jasper10x5dr

4 Phoneme-to-Grapheme Model

We seek a model for translating transcripts written in phonemes into graphemes in the same language. Unlike the most studies reviewed in Section 2, we propose to use Transformer (Vaswani et al., 2017) architecture for phoneme-to-grapheme translation. We believe that Transformer is the best option for these tasks. Transformer has shown its potential in many NLP tasks. Most importantly, we consider its ability to learn the structure of a sentence, see e.g. Pham et al. (2019).

4.1 Text Encoding Considerations

We use Byte Pair Encoding (BPE) (Sennrich et al., 2016) for text encoding in our experiments. We use the implementation in YouTokenToMe[6] library. It is fast and offers BPE-dropout (Provilkov et al., 2019) regularization technique.

First, we decided to use separate vocabularies for source and target sentences, because the source and target representations, IPA phonemes and English graphemes, have no substantial overlap.

There has been a quite intensive discussion about vocabulary size in neural machine translation (NMT) (Denkowski and Neubig, 2017; Gupta et al., 2019; Ding et al., 2019). All works agree that for low-resource translation tasks, it is better to apply smaller vocabulary sizes. For a high-resource task, it is convenient to use larger vocabulary. Our task, translation of phonemes into graphemes in the same language, differs from the previous works. Hence, we decided to experiment with vocabulary sizes. We also want to know whether we should train the sub-word units for source on clean data (phonemes without errors), or we should introduce ASR-like errors to these data.

We design the experiment as follows: we test character-level encoding and BPE vocabulary sizes of 128, 512, 2 000, 8 000 and 32 000. Further, we test a clean data configuration, "corrupted" data (we collect transcripts from an ensemble of 10 ASR systems) and a "mixed" data — combination of the two previous.

Because of the data scarcity, we use Transformer Base configuration. We alter maximum sequence length to 1024 because for character-level, 128, and 512 BPE configurations, many sentences do not fit into the model. We train all models for 70 000 steps on one GPU using the same batch size for all configurations: 12 000 tokens. We set the learning rate

[6] https://github.com/VKCOM/YouTokenToMe

193

to 0.04. As training data, we use "corrupted" ASR transcripts paired with true transcripts. We collect the data from an ensemble of 10 ASR models, yielding approximately 7 million sentence pairs. For the collection of ASR corrupted data, we used LibriSpeech and Common Voice datasets.

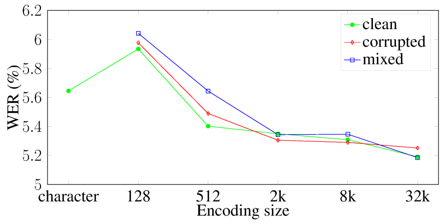

Figure 2: Results in % of word error rate on the Common Voice test set.

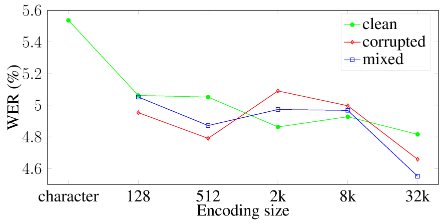

Figure 3: Results in % of word error rate on the LibriSpeech test clean.

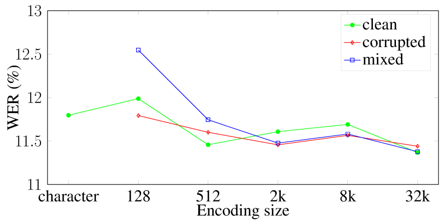

Figure 4: Results in % of word error rate on the LibriSpeech test other.

Graphical comparison is in Figures 2 to 4.

BPE size Character-level encoding seems to be the worst or second-worst possible representation. For the Common Voice test set, it scores almost one percentage point of WER more compared to the best result (5.53 vs 4.55). Also, all other encodings performed almost half a percentage point better. For both LibriSpeech test sets, it performed a bit better than BPE 128.

Generally, the results suggest a the larger the vocabulary, the lower WER. Among the different BPE sizes, we can recognize the 32 000 vocabulary size has the best results systematically on all test sets.

Finally, we consider the following: a model can better learn from larger vocabulary sizes. First, a model does not have to learn low-level orthography extensively. Rather than memorizing characters (or other smaller units), it can focus on the whole sentence and how individual words interact. Second, a larger model can detect errors because of anomalies in the input encoding. Larger vocabularies produce a shorter representation. Corrupted word is more likely to be broken down to smaller pieces. When a model detects such a situation, it can, for example, decide the right target word based on context, rather than the suspicious word. Such anomaly will most likely not occur in the text encoded with small BPE.

Source of BPE training data For Common Voice, we observe some variation in performance. Best seems to be the "mixed" configuration. Somewhat worse is "corrupted" and the worst is "clean" version. In this case, we think the "mixed" is best as it has frequent enough "corrupted" words. This enables a model to learn to translate these corrupted words into the correct ones. Also, it knows enough other words, so it can adequately work with correct phonemes.

For other test sets, we observe almost no differences. Only "corrupted" configuration has slightly worse performance.

We conclude that the source of training data for BPE has almost no impact on the final result.

4.2 Baseline Phoneme-to-Grapheme Model ("asr" Configuration)

We decided to use Transformer Big configuration (as opposed to the initial experiment with BPE vocabularies). As we concluded in the previous part, we select BPE vocabulary size of 32 000, and the BPE encoding is trained on "clean" phonemized English part of Czeng 1.7 (Bojar et al., 2016) corpus.

First, we train a randomly initialized Transformer model. The source of the "translation" is the phonemized English Czeng and the target is the original English.

194

We use six 16 GB GPUs for the training. We set the batch size to 6 000 tokens, learning rate to 0.02, warm-up steps to 16 000 and total steps to 600 000. We manually abort the training after the convergence is reached (140 000 steps in our case).

4.3 Transfer from SLT ("asr_slt" Configuration)

In standard NMT, the source text usually does not suffer from so many errors as in our setup. We address this "correction" need by training on artificially corrupted source side.

We initialize the Transformer encoder from our in-house speech translation model trained from English phonemes to Czech graphemes (described in Polák (2020)) and the decoder from a model for the opposite direction. Both of these initial models were trained on CzEng, with one side converted to phonemes using `phonemizer`.

These pre-trained parts of the model, the encoder and decoder need joint training to learn to operate with each other. We employ this training also to inject the capacity of correcting ASR output.

Specifically, we apply the jack-knife scheme to our ASR training data (LibriSpeech and Common Voice), training ten different ASR models, always leaving one-tenth of the training data aside. This one-tenth is recognized with the model, leading to the full speech corpus equipped not only with golden transcripts but also with ASR outputs. We call this an "ASR-corrupted" corpus.

Based on our experience from the experiment with BPE vocabularies, where the model easily over-fit to the sentences from ASR transcripts from speech corpora, we mix the corrupted and clean data with a 1:1 ratio. This is different from Hrinchuk et al. (2019) who use only the ASR-corrupted data to train. We then train the complete Transformer model from English phonemes to English graphemes with the same hyper-parameters as the baseline.

4.4 Transfer from BERT ("asr_bert" Configuration)

Finally, we use the pre-trained BERT (Devlin et al., 2018). Unlike Hrinchuk et al. (2019), we do not initialize both the encoder and decoder with the BERT. We initialize the encoder from the English-to-Czech speech translation model (as in Section 4.3) because we need the model to process phonemes, not graphemes on the source side. The decoder is initialized from the BERT "large" to match the dimension of the Transformer encoder.

For this setup, we tried the same training procedure on half-noisy data as above. However, we were unable to obtain any reasonable performance (we got WER of 28 % on LibriSpeech dev-other). We hypothesize this is due to the vast amount of weights that must be randomly initialized in the decoder: BERT is a Transformer encoder only. Hence it does not have the Encoder-Decoder attention layer which must be trained from scratch. During the training of the whole model with many randomly initialized weights, the initially trained weights from the BERT might depart too far from the optimum.

To overcome this issue, we use an analogous adaptation trick as for the training of the acoustic model. We freeze all weights initialized from seed models and train only the randomly initialized weights until convergence (the criterion was the loss on the validation dataset). This adaptation takes 13 500 steps in our case. Subsequently, the training continues as in the previous case with one exception — we used only ASR corrupted data from LibriSpeech.

4.5 ASR Results

	CV	LS clean	LS other
asr (primary)	9.72	4.87	11.67
asr_lm	7.00	4.63	10.25
asr_slt	**3.26**	5.10	11.75
asr_slt_lm	3.97	5.00	10.63
bert	12.93	4.13	10.21
bert_lm†	11.25	**4.04**	**9.69**

Table 2: Performance of the submitted models in terms of % WER on the Common Voice test set (CV), and LibriSpeech (LS) clean and other test set. † not submitted due to time constraints. Best results in bold.

Table 2 reports the performance of our proposed systems on Common Voice test set and LibriSpeech test-clean and test-other.

The performance of "slt"-pretrained models is very good on Common Voice (CV), reaching WER of 3.26 %. However, we suspect that the model overfitted to CV texts. The corpus contains many speakers, but the set of underlying sentences is very limited, and our models can memorize them. The more realistic evaluation on the independent LibriSpeech other indicates that "asr_slt" is actually rather poor.

For the general domain, assessed by LibriSpeech

	AMIa	AMIb	AMIc	AMId	Teddy	Autocentrum	Audit	Weighted AMI	Weighted Rest	Weighted Total
asr (primary)	**35.89**	**32.76**	35.60	39.90	57.43	11.62	9.83	**35.05**	19.67	**33.99**
asr_lm†	37.58	33.66	**35.32**	40.60	56.65	14.01	11.00	35.70	20.54	34.65
asr_slt	36.73	33.22	35.70	39.69	56.87	**10.93**	10.22	35.37	19.66	34.28
asr_slt_lm†	37.71	33.83	35.67	40.45	56.31	12.87	10.71	35.88	20.07	34.78
asr_bert	36.69	33.82	36.50	**39.63**	56.76	12.87	**9.60**	35.85	**19.64**	34.72
asr_lm‡	35.95	32.94	35.57	40.43	**56.20**	13.10	10.67	35.20	20.22	34.17
asr_slt_lm‡	37.72	33.86	35.59	40.59	56.42	13.10	10.71	35.88	20.29	34.80
Microsoft	53.72	52.62	56.67	58.58	87.82	39.64	24.22	54.80	39.75	53.76
Google	51.52	49.47	53.11	56.88	61.01	14.12	17.47	51.87	25.33	50.03

Table 3: Results in % WER on IWSLT ASR development set. † submitted without punctuation and segmentation. ‡ submitted with punctuation and segmentation after the deadline.

	AMIa	Teddy	Autocentrum	Audit
asr	4.79	1.41	21.66	5.59
asr_lm†	2.80	1.57	12.53	1.84
asr_lm†*	2.86	1.57	12.79	1.93
asr_slt	4.52	1.48	**22.02**	5.56
asr_slt_lm†	3.19	1.55	8.85	1.81
asr_slt_lm†*	3.26	1.55	9.32	1.88
asr_bert	**6.08**	1.41	19.01	**5.79**
asr_lm‡	3.92	5.65	21.65	5.24
asr_slt_lm‡	4.01	**6.08**	21.50	5.02
Gold	21.09	54.77	42.52	9.03

Table 4: Czech BLEU scores on the IWSLT development set. † submitted without punctuation and segmentation. ‡ submitted with punctuation and segmentation after the deadline. * lower case BLEU.

	AMIa	Teddy	Autocentrum	Audit
asr	8.87	**5.20**	15.94	22.40
asr_lm†	3.45	2.02	4.16	6.15
asr_lm†*	5.30	4.33	8.64	18.16
asr_slt	9.77	4.35	16.40	22.91
asr_slt_lm†	3.45	2.21	4.00	6.54
asr_slt_lm†*	5.34	4.20	6.92	20.07
asr_bert	10.22	3.99	13.38	24.76
asr_lm‡	10.79	4.36	17.24	25.09
asr_slt_lm‡	**10.88**	3.60	**17.34**	**26.64**
Gold	34.95	45.57	36.56	38.97

Table 5: German BLEU scores on the IWSLT development set. † submitted without punctuation and segmentation. ‡ submitted with punctuation and segmentation after the deadline. * lower case BLEU.

clean, we would choose the BERT-pretrained model with phoneme LM rescoring. This model was unfortunately trained too late, so we did not include it in our submission.

The Non-Native Task setting is very specific, and we carefully examine the performance on the IWSLT development (Table 3). The performance varies considerably, but the baseline setup (**"asr"**) perform well on average, and it is also not much worse than the best system on the particular files, e.g. 9.83 on the Audit file compared to "asr_bert" which wins there with 9.60. Based on these results, we selected **"asr"** as our primary submission for speech recognition track.

It the particular domain of non-native speech recognition, the usefulness of the phoneme language model seems to be minor, unlike on the CV and LS test sets in Table 2. However, this result could be unreliable because the IWSLT development set is very small.

We note that all proposed systems outperform publicly available Google and Microsoft ASR on all files in the development set, see the last two rows of Table 3.

5 Punctuation, Truecasing and Segmentation

Our ASR system produces lowercased, unpunctuated text, but the machine translation works on capitalized, punctuated text, segmented to individual sentences. We use the same biRNN punctuator, truecaser and segmenter as Macháček et al. (2020). The punctuator is a bidirectional recurrent neural network by Tilk and Alumäe (2016) trained on the English side of CzEng (Bojar et al., 2016). The truecaser uses tri-grams (Lita et al., 2003). We use a rule-based Moses Sentence Splitter (Koehn et al., 2007). More details are in Macháček et al. (2020), Section 4.2.

6 Machine Translation

Our submission to the SLT track relies on the MT systems, which are used also by ELITR project and are described in their submission to this task (Macháček et al., 2020). We do not rely on their validation for this task. As our primary MT systems, we select "WMT18 T2T" for Czech and "de T2T" for German, because they were easily accessible

196

Name	Initialization		LM rescoring
	Encoder	Decoder	
asr (primary)	random	random	no
asr_lm	random	random	yes
asr_slt	EN CS	CS EN	no
asr_slt_lm	EN CS	CS EN	yes
bert	EN CS	BERT	no

Table 6: Submitted English ASR configurations. "EN CS" means the Transformer encoder was initialized with the encoder weights from a translation model trained from English phonemes to Czech graphemes. "CS EN" means the decoder was initialized from an MT model translating Czech phonemes to English graphemes.

through Lindat service[7].

"WMT18 T2T" was originally trained for English-Czech WMT18 news translation task (Popel, 2018), and was also between the top systems in WMT19 (Popel et al., 2019). It is a single-sentence Transformer Big model in Tensor2Tensor framework (Vaswani et al., 2018). "de T2T" is a similar system, but trained on the data for English-German WMT news translation. Tables 4 and 5 present BLEU scores of our primary systems for Czech and German, respectively. Note that the files Teddy, Autocentrum and Audit are very short.

We submit also all other machine translation systems for Czech and German by ELITR with our "asr" source for contrastive evaluation. See Macháček et al. (2020) for more details.

7 Submission Summary

We participate in two tracks of the non-native speech translation task: speech recognition, and speech translation into both Czech and German. In both cases, our submissions are off-line.

The acoustic model was initialized from a checkpoint trained on other data than allowed for the task. Therefore, our systems are unconstrained.

For the speech recognition track, we utilize our speech recognition pipeline in various configurations. We first obtain the phoneme transcripts using the acoustic model. For configurations marked with "_lm", we additionally use a phoneme language model during the acoustic model inference. Subsequently, we feed these phonetic transcripts to the phoneme-to-grapheme translation model. We have three variants of this model: plain ("asr"), with pre-trained weights from SLT ("slt"), and with pre-trained weights from SLT for encoder and BERT for decoder ("bert"). In this manner, we yield five different configurations for submission (see Table 6). The transcripts are then punctuated and truecased. Based on the punctuation, we further segment the transcripts. Our primary submission for the ASR track is the "asr" system.

We do not have our own translation model. To participate in the translation track, we utilize the MT systems of the ELITR project, which are mostly Transformer neural models. We select as our primary submission the "asr" system.

[7] https://lindat.mff.cuni.cz/services/translation/

8 Conclusion

We presented our submissions to the Non-Native Speech Translation Task for IWSLT 2020.

For the non-native speech recognition, we proposed a pipeline that consists of an acoustic model and a phoneme-to-grapheme model. We demonstrated that the proposed pipeline surpasses commercially used ASR on the development set.

To participate in the non-native speech translation track, we use off-the-shelf translation model on our ASR transcripts.

Acknowledgments

The research was partially supported by the grants 19-26934X (NEUREM3) of the Czech Science Foundation, H2020-ICT-2018-2-825460 (ELITR) of the EU, 98120 of the Grant Agency of Charles University and by SVV project number 260 575.

References

Ebrahim Ansari, Amittai Axelrod, Nguyen Bach, Ondrej Bojar, Roldano Cattoni, Fahim Dalvi, Nadir Durrani, Marcello Federico, Christian Federmann, Jiatao Gu, Fei Huang, Kevin Knight, Xutai Ma, Ajay Nagesh, Matteo Negri, Jan Niehues, Juan Pino, Elizabeth Salesky, Xing Shi, Sebastian Stüker, Marco Turchi, and Changhan Wang. 2020. Findings of the IWSLT 2020 Evaluation Campaign. In *Proceedings of the 17th International Conference on Spoken Language Translation (IWSLT 2020)*, Seattle, USA.

Willem D Basson and Marelie H Davel. 2013. Category-based phoneme-to-grapheme transliteration.

Ondřej Bojar, Ondřej Dušek, Tom Kocmi, Jindřich Libovický, Michal Novák, Martin Popel, Roman Sudarikov, and Dušan Variš. 2016. CzEng 1.6: Enlarged Czech-English Parallel Corpus with Processing Tools Dockered. In *Text, Speech, and Dialogue: 19th International Conference, TSD 2016*, number

9924 in Lecture Notes in Computer Science, pages 231–238, Cham / Heidelberg / New York / Dordrecht / London. Masaryk University, Springer International Publishing.

B. Decadt, J. Duchateau, W. Daelemans, and P. Wambacq. 2001. Phoneme-to-grapheme conversion for out-of-vocabulary words in large vocabulary speech recognition. In *IEEE Workshop on Automatic Speech Recognition and Understanding, 2001. ASRU '01.*, pages 413–416.

Michael Denkowski and Graham Neubig. 2017. Stronger baselines for trustable results in neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 18–27.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Shuoyang Ding, Adithya Renduchintala, and Kevin Duh. 2019. A call for prudent choice of subword merge operations in neural machine translation. In *Proceedings of Machine Translation Summit XVII Volume 1: Research Track*, pages 204–213.

Rohit Gupta, Laurent Besacier, Marc Dymetman, and Matthias Gallé. 2019. Character-based nmt with transformer. *arXiv preprint arXiv:1911.04997*.

Axel Horndasch, Elmar Nöth, Anton Batliner, and Volker Warnke. 2006. Phoneme-to-grapheme mapping for spoken inquiries to the semantic web. In *Ninth International Conference on Spoken Language Processing*.

Oleksii Hrinchuk, Mariya Popova, and Boris Ginsburg. 2019. Correction of automatic speech recognition with transformer sequence-to-sequence model. *arXiv preprint arXiv:1910.10697*.

Biing-Hwang Juang and Lawrence R Rabiner. 2005. Automatic speech recognition–a brief history of the technology development. *Georgia Institute of Technology. Atlanta Rutgers University and the University of California. Santa Barbara*, 1:67.

Philipp Koehn et al. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *ACL Interactive Poster and Demonstration Sessions*, page 177–180, USA. Association for Computational Linguistics.

Oleksii Kuchaiev, Jason Li, Huyen Nguyen, Oleksii Hrinchuk, Ryan Leary, Boris Ginsburg, Samuel Kriman, Stanislav Beliaev, Vitaly Lavrukhin, Jack Cook, Patrice Castonguay, Mariya Popova, Jocelyn Huang, and Jonathan M. Cohen. 2019. Nemo: a toolkit for building ai applications using neural modules.

Jason Li, Vitaly Lavrukhin, Boris Ginsburg, Ryan Leary, Oleksii Kuchaiev, Jonathan M. Cohen, Huyen Nguyen, and Ravi Teja Gadde. 2019. Jasper: An End-to-End Convolutional Neural Acoustic Model. In *Proc. Interspeech 2019*, pages 71–75.

Lucian Vlad Lita, Abe Ittycheriah, Salim Roukos, and Nanda Kambhatla. 2003. Truecasing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, page 152–159, USA. Association for Computational Linguistics.

Dominik Macháček, Jonáš Kratochvíl, Sangeet Sagar, Matúš Žilinec, Ondřej Bojar, Thai-Son Nguyen, Felix Schneider, Philip Williams, and Yuekun Yao. 2020. ELITR Non-Native Speech Translation at IWSLT 2020. In *Proceedings of the 17th International Conference on Spoken Language Translation (IWSLT 2020)*, Seattle, USA.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210. IEEE.

Thuong-Hai Pham, Dominik Macháček, and Ondřej Bojar. 2019. Promoting the knowledge of source syntax in transformer nmt is not needed. *Computación y Sistemas*, 23(3):923–934.

Peter Polák. 2020. Spoken language translation via phoneme representation of the source language. Master's thesis, Charles University.

Martin Popel. 2018. CUNI transformer neural MT system for WMT18. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 482–487, Belgium, Brussels. Association for Computational Linguistics.

Martin Popel, Dominik Macháček, Michal Auersperger, Ondřej Bojar, and Pavel Pecina. 2019. English-czech systems in wmt19: Document-level transformer. In *WMT*.

Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. 2019. Bpe-dropout: Simple and effective subword regularization. *arXiv preprint arXiv:1910.13267*.

Michael D Riley and Andrej Ljolje. 1992. Recognizing phonemes vs. recognizing phones: a comparison. In *Second International Conference on Spoken Language Processing*.

Elizabeth Salesky, Matthias Sperber, and Alan W Black. 2019. Exploring phoneme-level speech representations for end-to-end speech translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1835–1841, Florence, Italy. 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), pages 1715–1725.

Ottokar Tilk and Tanel Alumäe. 2016. Bidirectional recurrent neural network with attention mechanism for punctuation restoration. pages 3047–3051.

Ashish Vaswani, Samy Bengio, Eugene Brevdo, Francois Chollet, Aidan N. Gomez, Stephan Gouws, Llion Jones, Łukasz Kaiser, Nal Kalchbrenner, Niki Parmar, Ryan Sepassi, Noam Shazeer, and Jakob Uszkoreit. 2018. Tensor2tensor for neural machine translation. *CoRR*, abs/1803.07416.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Alex Waibel, Toshiyuki Hanazawa, Geoffrey Hinton, Kiyohiro Shikano, and Kevin J Lang. 1989. Phoneme recognition using time-delay neural networks. *IEEE transactions on acoustics, speech, and signal processing*, 37(3):328–339.