Charles University in Prague

Faculty of Mathematics and Physics

BACHELOR THESIS

Ondřej Cífka

Machine Translation of Spoken English into Czech

Institute of Formal and Applied Linguistics

Supervisor of the bachelor thesis:	RNDr. Ondřej Bojar, Ph.D.
Study programme:	Computer Science
Study branch:	General Computer Science

Prague 2016

I declare that I carried out this bachelor thesis independently, and only with the cited sources, literature and other professional sources.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University in Prague has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In .... date .... signature of the author

Název práce: Strojový překlad z mluvené angličtiny do češtiny

Autor: Ondřej Cífka

Ústav: Ústav formální a aplikované lingvistiky

Vedoucí bakalářské práce: RNDr. Ondřej Bojar, Ph.D., Ústav formální a aplikované lingvistiky

Abstrakt: Automatický překlad mluvené řeči z jednoho jazyka do druhého se v dnešní době stává žádaným nástrojem k překonání jazykové bariéry. Tato práce se věnuje strojovému překladu mluvené angličtiny do češtiny jakožto pomůcce pro zahraniční turisty. Nejprve jsme z volně dostupných součástí sestavili plně funkční překladový systém a s jeho pomocí nasbírali od uživatelů ukázkové vstupy. Poté jsme se zaměřili na dvě nejdůležitější součásti systému – automatické rozpoznání řeči (ASR) a strojový překlad (MT) – a pokusili se je nahradit vlastními modely, přizpůsobenými pro danou doménu. Nakonec jsme tato vylepšení vyhodnotili na nasbíraných datech.

Klíčová slova: překlad mluvené řeči, rozpoznání řeči, strojový překlad, čeština, angličtina

Title: Machine Translation of Spoken English into Czech

Author: Ondřej Cífka

Institute: Institute of Formal and Applied Linguistics

Supervisor: RNDr. Ondřej Bojar, Ph.D., Institute of Formal and Applied Linguistics

Abstract: Spoken language translation, the process of translating speech in one language into another language automatically, is in increasing demand as a means of overcoming the language barrier. In this thesis, we focus on translation of spoken English into Czech, employed as an aid for international tourists. We built a fully functional speech translation system using freely available components and used it for collecting samples of user input. We then focused on replacing the core components of the system, namely speech recognition (ASR) and machine translation (MT), with our own, domain-adapted models. We evaluated our improvements on the collected data.

Keywords: spoken language translation, speech recognition, machine translation, Czech, English

I would like to thank RNDr. Ondřej Bojar, Ph.D., Filip Jurčíček, Ph.D. and Mgr. Ondřej Plátek for their guidance and help. My gratitude also goes to my girlfriend Kateřina Apolínová for her support and for her assistance with preparing the test data.

Contents

Introduction

Spoken language translation

The goal of spoken language translation (SLT) is to process recorded speech in one language to produce a translation in another language. In the recent years, SLT has become increasingly popular both in research and commercial applications. Since 2015, the Google Translate mobile app features near-simultaneous translation of bilingual conversations, and the same functionality is offered by Skype Translator. In this thesis, we attempt a similar feat for the English-Czech language pair only. We present a telephone-based speech translation system designed to help foreign tourists in the Czech Republic. It can be thought of as a sort of interactive phrasebook, and its domain is therefore somewhat limited.

A speech translation system typically deals with two major subtasks: (automatic) speech recognition (ASR), i.e. the translation of the audio signal to a sequence of words in the source language, and machine translation (MT), i.e. the translation of this sequence of words into the target language. The aim of this thesis was to build a prototype of such a system, using existing components for solving these subtasks, and then build our own models for ASR and MT, adapted to the 'travel domain' and designed to eventually replace the ones used in the prototype.

Structure of the thesis

This thesis is organized as follows. In Chapter 1, we will give an account of the theoretical background of spoken language translation, then describe our prototype of an interactive speech-to-speech translation system and analyse its output on the data it collected. In Chapter 2, we shall walk through the design of our experimental system and the process of building its individual components. Finally, in Chapter 3, we will evaluate the performance of different variations of this system on the collected data.

1. Analysis

1.1 Background

Before we delve into the details of our experimental system, let us give a theoretical overview of some of the techniques used in speech recognition and machine translation, and present related work on spoken language translation.

1.1.1 Statistical speech recognition

The role of a speech recognizer is to translate a speech signal to a string of words. In statistical speech recognition, the signal is represented as a sequence of acoustic feature vectors $\mathbf{x}$ and we want to find the most likely word sequence $\hat{\mathbf{w}}$ given $\mathbf{x}$:

$$\hat{\mathbf{w}} = \arg\max_{\mathbf{w}} p(\mathbf{w} \mid \mathbf{x}). \tag{1.1}$$

According to Bayes' theorem:

$$p(\mathbf{w} \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid \mathbf{w})p(\mathbf{w})}{p(\mathbf{x})}. \tag{1.2}$$

Since the denominator is a positive constant in the maximization, the equation (1.1) becomes

$$\hat{\mathbf{w}} = \arg\max_{\mathbf{w}} p(\mathbf{x} \mid \mathbf{w})p(\mathbf{w}), \tag{1.3}$$

which is sometimes called the fundamental equation of speech recognition. In modern ASR systems, the probability $p(\mathbf{w})$ of a sequence of words is modelled by the *language model* (LM), while the conditional probability $p(\mathbf{x} \mid \mathbf{w})$ is computed using the *acoustic model* (AM). The component connecting these two statistical models is the pronunciation lexicon, which describes the pronunciation of each word using a sequence of phones. The acoustic model then represents the distributions of acoustic features for individual phones.

The details of acoustic modelling will not be discussed here since training an acoustic model is not a subject of this thesis. We will elaborate on language modelling in Section 1.1.3.

1.1.2 Statistical phrase-based machine translation

Statistical phrase-based machine translation [16] is one of the most popular approaches to machine translation today and, despite its apparent simplicity, also one of the most effective. Other approaches like rule-based translation and statistical syntax-based translation can in some cases perform comparably well but they require extensive linguistic knowledge and usually underperform the phrase-based approach in unrestricted domains or with noisy input, as can be expected from ASR.

The task of machine translation is analogous to that of speech recognition: instead of a sound signal, we have a string of words in the source language that we want to translate to a string of words in the target language. In other words, we would like to find the most likely Czech translation $\hat{\mathbf{e}}$ of a given English sentence $\mathbf{f}$:

$$\hat{\mathbf{e}} = \arg\max_{\mathbf{e}} p(\mathbf{e} \mid \mathbf{f}). \tag{1.4}$$

We can then derive the analogue of the equation (1.3) for machine translation:

$$\hat{\mathbf{e}} = \arg\max_{\mathbf{e}} p(\mathbf{f} \mid \mathbf{e}) p(\mathbf{e}). \tag{1.5}$$

This equation captures the *noisy channel model*, which is the basis of statistical machine translation. Again, we use two separate statistical models to represent these probabilities: the language model $p(\mathbf{e})$, modelling the probabilities of Czech sentences, and the *translation model* (TM) $p(\mathbf{f} \mid \mathbf{e})$, describing how likely the English sentence $\mathbf{f}$ is a translation of the Czech $\mathbf{e}$. The usefulness of this approach lies in the fact that the language model can act as a correction mechanism for the translation model, assigning higher probabilities to sentences that sound more correct (natural, fluent) in Czech.

A generalization of the noisy channel model is the *log-linear model* [23], which allows to replace the language model and the translation model with an arbitrary set of *feature functions*. The probability $p(\mathbf{e} \mid \mathbf{f})$ is modelled as

$$p(\mathbf{e} \mid \mathbf{f}) = \frac{\exp\left(\sum_{m=1}^{M} \lambda_m h_m\left(\mathbf{e}, \mathbf{f}\right)\right)}{\sum_{\mathbf{e}'} \exp\left(\sum_{m=1}^{M} \lambda_m h_m\left(\mathbf{e}', \mathbf{f}\right)\right)}, \tag{1.6}$$

where $h_1, h_2, \ldots, h_M$ are the feature functions and $\lambda_1, \lambda_2, \ldots, \lambda_M$ their respective weights. Plugging in to equation (1.5) gives (dropping once again the positive constant denominator):

$$\begin{aligned}
\hat{\mathbf{e}} &= \arg\max_{\mathbf{e}} \frac{\exp\left(\sum_{m=1}^{M} \lambda_m h_m(\mathbf{e}, \mathbf{f})\right)}{\sum_{\mathbf{e}'} \exp\left(\sum_{m=1}^{M} \lambda_m h_m(\mathbf{e}', \mathbf{f})\right)} \\
&= \arg\max_{\mathbf{e}} \exp\left(\sum_{m=1}^{M} \lambda_m h_m(\mathbf{e}, \mathbf{f})\right) \\
&= \arg\max_{\mathbf{e}} \sum_{m=1}^{M} \lambda_m h_m(\mathbf{e}, \mathbf{f}).
\end{aligned} \tag{1.7}$$

We can see that if we choose $M = 2$, $h_1(\mathbf{e}, \mathbf{f}) = \log p(\mathbf{f}|\mathbf{e})$, $h_2(\mathbf{e}, \mathbf{f}) = \log p(\mathbf{e})$ and $\lambda_1 = \lambda_2 = 1$, we obtain the noisy channel model. Another example of a possible feature function is the word penalty, which is simply the length of the translation; the weight assigned to it can be used to control the length of the sentences produced by the system.

Translation model

Phrase-based translation works by splitting up the sentence into segments called *phrases*, translating each phrase separately and then putting them back together (not necessarily in the original order) to form the target sentence. There are many ways to split up the sentence and many translation options for each phrase; the choice depends on the probabilities assigned by the log-linear model.

An indispensable feature in the log-linear model is of course the translation model. Its basis is the *phrase table*, which lists the possible translations of each phrase and their probabilities. The model is trained on a parallel corpus, i.e. a large sentence-aligned collection of texts in the source language and the target language. Phrases and their translations are extracted from this corpus and their probabilities are determined using the *maximum likelihood estimate* (MLE). For phrase extraction, the alignment of individual words is usually a prerequisite.

Weight tuning

The log-linear model comes with the need to determine the feature weights $\lambda_1, \lambda_2, \ldots, \lambda_M$. The standard method for weight tuning used in statistical machine translation is the Minimum Error Rate Training (MERT) [22], an algorithm which optimizes the weights toward an automatic metric of translation quality, such as BLEU (see 1.1.4).

To use MERT, we need another parallel corpus: the so-called *tuning* or *development set*. Roughly speaking, the tuning consists of the following steps:

1. Translate each sentence in the tuning set using the MT model to obtain an n-best list (a list of n best translations according to the model).
2. Score each translation using the given metric.
3. Adjust the feature weights so that the best-ranked translations receive higher probabilities from the model.
4. Repeat from step 1 with the modified feature weights.

The process is repeated until the weights converge.

1.1.3 Language modelling

The role of language modelling is to estimate the probability $p(\mathbf{w})$ of a string of words $\mathbf{w} = w_1 w_2 \ldots w_k$. The usual approach is the *n-gram model*, which is based on the assumption that the conditional probability of the next word in a sequence depends only on the $n-1$ previous words (this is known as the *Markovian property*). The constant n is called the *order* of the model; common choices are $n = 2, 3, 4, 5$.

By applying the chain rule, we can express the probability $p(\mathbf{w})$ as follows:[1]

$$\begin{aligned} p(\mathbf{w}) &= p(w_1 w_2 \ldots w_k) \\ &= p(w_1) \cdot p(w_2 \mid w_1) \cdot p(w_3 \mid w_1 w_2) \cdots p(w_k \mid w_1 w_2 \ldots w_{k-1}). \end{aligned} \tag{1.8}$$

Now we can use the Markovian property to approximate the probability:

$$p(\mathbf{w}) \cong \prod_{i=1}^{k} p(w_i \mid w_{i-n+1} \ldots w_{i-2} w_{i-1}). \tag{1.9}$$

[1] Here, $p(w_k \mid w_1 w_2 \ldots w_{k-1})$ denotes the conditional probability of w_k given the $k-1$ preceding words.

These conditional probabilities can be estimated from a text corpus using MLE by counting the number of times each n-gram (sequence of n words) and $(n-1)$-gram occurs in the corpus:

$$p(w_n \mid w_1 w_2 \dots w_{n-1}) = \frac{\text{count}(w_1 w_2 \dots w_{n-1} w_n)}{\text{count}(w_1 w_2 \dots w_{n-1})}. \tag{1.10}$$

This straightforward approach has an obvious drawback: if a sentence contains an n-gram that was never seen in the training data, the sentence will receive a zero probability from the LM. However, such a sentence can still be perfectly correct. It is therefore important to apply *smoothing*, which adjusts the way n-gram probabilities are calculated so that non-zero probabilities are assigned to unseen n-grams. A variety of smoothing techniques exists; in this thesis, we use the modified Kneser-Ney smoothing [7], which is probably the most popular approach.

1.1.4 Evaluation metrics

The performance of MT and ASR systems is often evaluated automatically in terms of how close the output of the system (the *candidate*) is to the desired output (*reference*). At least for MT, the ultimate measure of quality is always human judgement, but for many uses, it is too expensive and time-consuming. Automatic metrics constitute a much cheaper and faster, though somewhat less reliable alternative. They are particularly useful in situations where we need to repeatedly gauge the performance of a system on the same set of sentences, such as in parameter tuning. Two such commonly used metrics are WER and BLEU.

WER

The *word error rate* (WER) [34] metric is commonly used for evaluating speech recognition systems. For a single sentence, it is computed as the edit distance (or Levenshtein distance) between the reference and the candidate, normalized over the length of the reference. For a collection of sentences s_i with references r_i, we sum up all the respective edit distances and normalize over the total length of all references, i.e.

$$\text{WER} = \frac{\sum \min_{e \in E_i} (I(e) + D(e) + S(e))}{\sum |r_i|}.$$

E_i is the set of all editation sequences that transform r_i into s_i, and $I(e)$, $D(e)$ and $S(e)$ are the number of insertions, deletions and substitutions in a given editation sequence e, respectively. $|r_i|$ denotes the length of the sentence r_i.

BLEU

BLEU [26] is a standard evaluation metric for machine translation, based on n-gram precisions. It is defined as

$$\text{BLEU} = \text{BP} \cdot \exp\left(\frac{1}{4} \sum_{n=1}^{4} \log p_n\right)$$

where p_n is the modified n-gram precision

$$p_n = \frac{\text{number of } n\text{-grams shared with the reference}}{\text{total number of produced } n\text{-grams}}$$

and BP is the brevity penalty factor, penalizing candidates for being too short.

The best possible BLEU score is therefore 1 (or 100 when expressed as a percentage), but this value is by far not achievable in practice due to the existence of an overwhelming number of correct translations of each sentence.

1.1.5 Related work

ASR-MT interface

A frequent approach in speech translation is to have completely isolated ASR and MT systems that are connected serially, i.e. the latter processes the output of the former. This output can be in the form of a single (best-ranked) hypothesis, a list of hypotheses (an n-best list) or a word lattice (a compact representation of an n-best list) [27][6][19]. Alternatively, the two components can be more tightly integrated by using a common framework, e.g. a finite state transducer (FST) [5]. For simplicity, we take the first approach, passing a single hypothesis between the components.

Punctuation handling

As we will see, it is crucial that the MT model be trained on data that has the same characteristics as the expected input. A major issue in this regard is that the training data for MT models typically contains punctuation, whereas the output of ASR usually doesn't. Moreover, SLT is usually evaluated against a punctuated reference, and is therefore expected to produce punctuation. For these reasons, punctuation prediction, i.e. the automatic insertion of punctuation into a string of words, receives a lot of attention in SLT research. In general, three different approaches exist [20]:

- predict punctuation in the source language, i.e. before translation; the MT model is trained on punctuated data.
- remove punctuation from the source language training data but leave it in the target language data, leading to implicit punctuation prediction during translation.
- predict punctuation after translation; punctuation marks are removed from all training data.

In our case, however, we would eventually like to use the output of MT as input for speech synthesis, and therefore punctuation is arguably less important. Also, in our system, the input sound signal is divided into shorter segments before being processed by ASR, and we assume one sentence per segment; hence, punctuation prediction is not needed as a means of sentence segmentation.

We thus chose the path that seemed most straightforward: we follow the third approach with the difference that we do not predict punctuation at all.

As a result, our system produces unpunctuated output. Our test recordings were transcribed according to guidelines used in dialogue systems (see Sections 1.2 and 2.2.2 for details) and directly translated into Czech to obtain reference translations, and consequently these do not contain punctuation either.

Foreign words recognition

There has been a number of attempts at speech recognition including foreign words. A rather naïve method is to ignore the phonetic aspect of the foreign language altogether and simply generate the pronunciations of the foreign words using the grapheme-to-phoneme rules of the primary language (if these are available); for certain domains and language pairs (e.g. Sepedi with English words [21]), this can be surprisingly effective. An opposite approach is to use a specialized acoustic model (together with an appropriate pronunciation dictionary) – for example, we can build separate acoustic models for the two languages and then combine them into one model while merging similar phones from the two models; this approach has been tested on multiple language pairs including German-English [31] and Sepedi-English [21].

A kind of compromise between these two methods has been used, for example, for recognition of English words in Mandarin speech [35]. Here, only a Mandarin acoustic model is used and a set of rules is devised for converting English words to Mandarin phonetic representations, which are added to the pronunciation lexicon. In the present work, we adopt this last approach because it is better linguistically motivated than the first one, and doesn't require the acoustic model to be retrained, unlike the second presented method.

1.2 System prototype

As a first step, we implemented a prototype of an interactive English-to-Czech speech translation system.

We have discussed speech recognition and machine translation, the two major tasks in SLT. In an interactive speech-to-speech translation system, additional components are required. First of all, if the audio signal from the user is continuous, it usually needs to be pre-processed in order to identify segments of continuous speech – this process is called voice activity detection (VAD). As the last step of the process, speech in the target language needs to be synthesized from the translated text using a text-to-speech (TTS) engine.

We built such a system from existing components using the Alex open-source dialogue systems framework [14]. Alex provides the necessary tools to build a generic dialogue system; an example of its application is the Public Transit Information System [10].

Alex comes with a neural network-based voice activity detector, which we used in our prototype. It also supports different ASR and TTS engines; we employed the unofficial Google ASR web service[2] for English speech recognition, and a Czech TTS engine provided by SpeechTech.[3] For the MT subtask, we used the

[2] https://github.com/gillesdemey/google-speech-v2

[3] http://www.speechtech.cz/

Reference	**ASR output**
could you call me a taxi	you call me a taxi
he headed for the beer tent	for the beer tent
good morning what's the weather in the city of ústí nad labem	the weather in the city of Austin 11
does the tram twenty two go from here	the French wanted to go for my ear
there is a cake on the table	Ghost Recon trailer

Figure 1.1: Examples of errors in the ASR output. Differences between the output and the reference are underlined where possible.

phrase-based component of the hybrid system Chimera [2], available as a web service.[4]

The described system, named Alex Translate,[5] is being maintained as a phone-based service. Its source code is available on GitHub as a fork of the main Alex repository.[6] Its purpose was to collect samples of speech in our domain of interest, and this data was used in our experiments.

Below, we examine a sample of the collected sentences and the errors the system prototype made on this sample. More details about the data collected using the service will be given in Section 2.2.2.

1.2.1 ASR output

We will first inspect the output of the ASR component. Google ASR achieved a word error rate (WER) of 21.17 on the sample, with deletions and substitutions being most frequent. Indeed, missing and misrecognized words seem to be the most common problem in the output. In a few cases, no transcription was returned at all.

Examples of poorly recognized sentences are shown in Fig. 1.1. We can notice that words are often missing from the beginning of the sentence, probably disregarded as background noise by the model. These errors might be related to the fact that the majority of our speakers were non-native, while presumably Google ASR was trained mostly on data from native English speakers.

Rather unsurprisingly, the model was also unable to recognize the name of the Czech city *Ústí nad Labem*. It cannot be expected from a general-purpose ASR system for English to accurately recognize foreign words; this capability, however, seems desirable in an SLT application such as the one we are considering, where the users might occasionally use Czech words, especially place names.

1.2.2 MT output

We also examined the output of the system as a whole, i.e. the machine translation of the sentences produced by ASR. Many of the translation errors we found were

[4] `https://lindat.mff.cuni.cz/services/moses/`

[5] `https://ufal.mff.cuni.cz/alex#alex-translate`

[6] `https://github.com/cifkao/alex/tree/translate`

in fact introduced by speech recognition; we focused on the sentences that were recognized correctly, so as to discover some of the errors specific to the MT component, Chimera.

We identified various kinds of errors, both in fluency (wrong word order, bad word forms) and accuracy (wrong lexical choice, untranslated words). We found, however, that some of these errors could be at least partially corrected by simply adding punctuation to the source sentence. In the following examples, (a) is the machine translation of the raw ASR output and (b) denotes the machine translation of the same sentence with punctuation added:[7]

(1) where is the nearest tram stop [?]

(a) *kde je nejbližší tramvaj zastavit* ✗
where is the nearest tram$^{\text{N}}$ to stop

(b) *Kde je nejbližší tramvajová zastávka?* ✓
Where is the nearest tram$^{\text{Adj}}$ stop?

(2) nice to meet you [.]

(a) *těší vás* ✗
pleases you$_{acc.}$

(b) *Rád vás poznávám.* ✓
Pleased you$_{acc.}$ I meet.
'Pleased to meet you.'

This illustrates the mismatch between the two components: apparently, Chimera relies heavily on punctuation in the input since it was present in its training data; the output of the ASR component, on the other hand, is always unpunctuated.

[7] The square brackets indicate the punctuation that was added to the input for the purpose of this demonstration. Each translation is provided with a word-by-word gloss.

2. System design and training

This chapter describes in detail the design of our system and the training of the ASR and MT models.

2.1 Overview

A great amount of data is required for training both speech recognition and machine translation models. Unfortunately, our in-domain data is scarce: we have only collected about 14 minutes of speech, consisting of mere hundreds of sentences. Moreover, some data is needed for the evaluation of the system.

Since our collected in-domain data is insufficient for model training, we only use it for parameter tuning; the models themselves are trained on data from various other sources.

For speech recognition, we use an approach that allows us to bootstrap a domain-specific language model with little or no in-domain data available. We mix this LM with other LMs from different domains and combine it with an existing domain-independent acoustic model trained on both native and non-native English speech, which should be suitable for the travel domain.

For machine translation, we do not use any data tailored to our specific needs, but rely instead on the tuning of model parameters as a means of domain adaptation. We also seek to improve translation quality by ensuring uniform, 'speech-friendly' pre-processing for all training data.

Finally, we address the need for recognizing certain Czech words in English sentences by adding their pronunciations to the lexicon.

2.1.1 Tools

The training and usage of our system is illustrated in Fig. 2.1. The individual components are built using different sets of tools.

For training the language models, we use the SRI Language Modeling Toolkit (SRILM) [33]. SRILM allows not only for estimating language models from data, but also manipulating them, e.g. combining them using interpolation or pruning them to reduce their size.

The ASR model is built using the Kaldi toolkit [29]. Speech recognition itself is performed using an extension of the toolkit [28] which supports on-line recognition. We access the recognizer by means of the interface provided by the Alex framework.

We train and optimize the model for MT using the Moses statistical machine translation toolkit [17], with word alignment performed by GIZA++ [24]. The Moses decoder is used for translating.[1]

We build all of our models and conduct our experiments within the framework of Eman [1], an experiment manager. Intended for experiments in statistical

[1] In both ASR and MT, the term *decoding* refers to the search for the best hypothesis in the space of all possible hypotheses, or a bit more generally, to the process of speech recognition or machine translation itself (as opposed to the training phase).

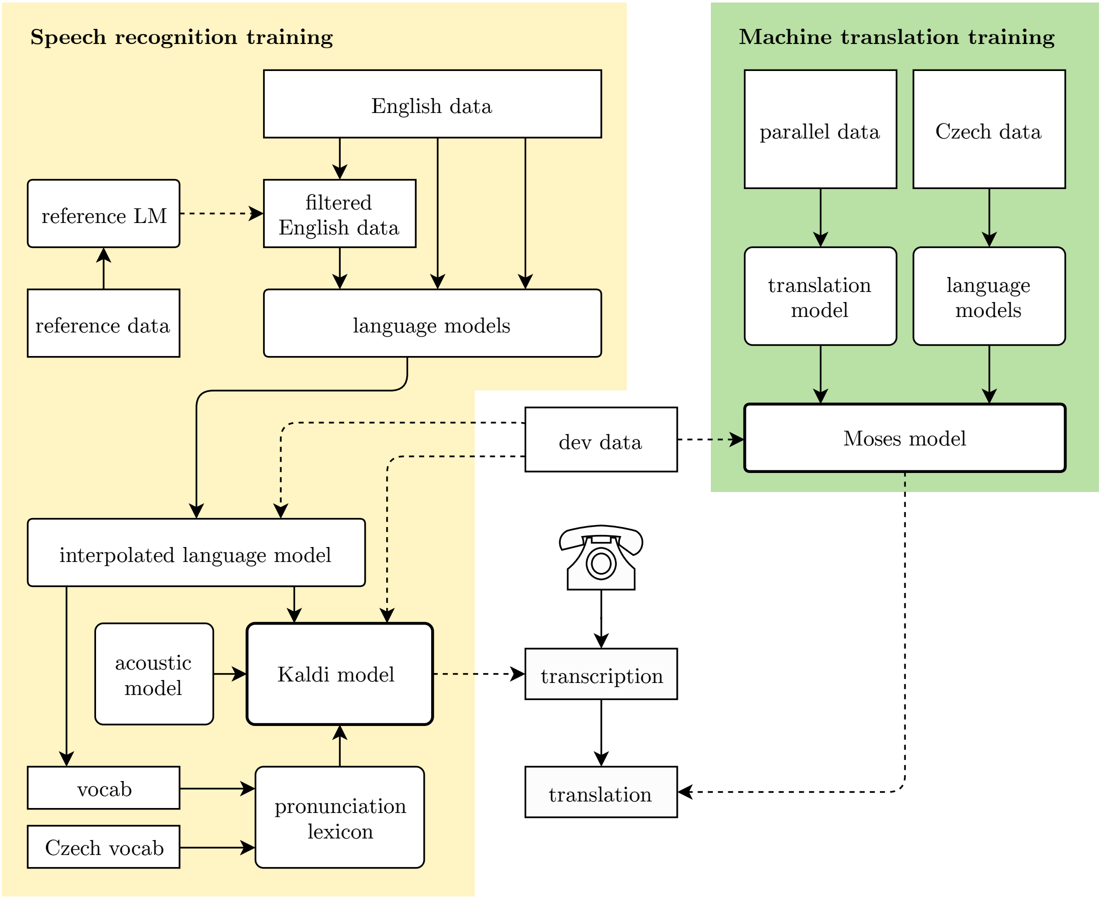


Figure 2.1: The training and application of the SLT system.

machine translation, Eman (or more precisely, the UFAL SMT Playground[2]) is bundled with wrapper scripts around toolkits like Moses and GIZA++ and other useful tools. Our copy of the UFAL SMT Playground is attached to this thesis so that the reader can examine our experiments for themselves (see Appendix A).

2.2 Corpora

The first step in the process of training the models is preparing the training data. For speech recognition, we use the following English corpora:

- Fiction and Subtitles from the CzEng corpus [3], version 1.6pre,
- a selection from the Common Crawl corpus [4],
- a selection of articles from Wikipedia.[3]

For machine translation, we use the entire deduplicated CzEng 1.6pre parallel corpus and the following Czech corpora:

- Fiction and Subtitles from the CzEng 1.6pre corpus,
- News Crawl articles from 2013 and 2014.[4]

Below, we go over the process of obtaining these corpora, as well as additional corpora used for parameter tuning and testing. For clarity, we include an overview of all these corpora in Table 2.1.

2.2.1 Speech normalization

We pre-process all the corpora using our 'speech normalization' script (see Appendix A.2.1). For every sentence in the corpus, we apply the following procedure:

1. Lowercase the sentence.
2. Normalize whitespace, i.e. replace every sequence of whitespace characters with a single space and remove leading and trailing whitespace.
3. Replace words like *uh* and *er* with the hesitation symbol `_EHM_HMM_`. This is because hesitation sounds, background noise and other non-speech events are treated as special kinds of phones in the acoustic model.
4. Convert numbers to their textual representation. This is done by searching for numbers using a regular expression and then replacing the matches with the help of the Perl modules `Lingua::EN::Numbers`[5] for English and `Lingua::CS::Num2Word`[6] for Czech. (If the module in use fails to convert a number, we remove it from the text.)

[2] `https://redmine.ms.mff.cuni.cz/projects/ufal-smt-playground`

[3] `https://en.wikipedia.org/`

[4] `http://www.statmt.org/wmt14/translation-task.html#download`

[5] `http://search.cpan.org/~neilb/Lingua-EN-Numbers-2.03/lib/Lingua/EN/Numbers.pm`

[6] `http://search.cpan.org/~rvasicek/Lingua-CS-Num2Word-0.03/Num2Word.pm`

	name	sentences	words	used for…
English	*c-fiction*	6 M	77 M	ASR LMs
	c-subtitles	36 M	280 M	
	cc-select-60	29 M	187 M	
	cc-select-80	46 M	347 M	
	cc-select-100	66 M	558 M	
	wiki-neighbors	810 k	19 M	
	lmppl-ref	388	1.4 k	*cc-select-∗* filtering
	mixlm-tuning	133	709	ASR LM interpolation
	calls1	176	725	ASR tuning
	cstest	256	1.8 k	Czech words ASR evaluation
Czech	*c-fiction*	6 M	67 M	MT LMs
	c-subtitles	36 M	227 M	
	newscrawl	18 M	251 M	
parallel	*czeng16*	51 M	584 M	translation model
	mert-tuning	414	1.8 k	MT tuning
	calls2	188	787	evaluation

Table 2.1: A summary of the used corpora. Word and sentence counts refer to the normalized versions. For parallel corpora, English word counts are shown.

5. Remove all non-word characters except for apostrophes used within words (such as *don't*, *she's*).

6. Discard the sentence if it is empty.

This effectively tokenizes the sentence, turning it into a sequence of lowercase words separated by spaces.

2.2.2 Call transcriptions

We used our system prototype to collect an English speech corpus, consisting of call recordings (segmented into utterances by the Alex voice activity detector) and their manual transcriptions. The annotators adhered to the instructions for phone call transcription used in the Alex framework. In particular, no punctuation was included and all non-speech events (hesitation sounds, laughing, breathing, background noises) were transcribed using special symbols. Recordings containing unintelligible speech or speech in a language other than English were excluded from the corpus. The appropriate transcriptions were translated into Czech (while retaining the non-speech event symbols) to obtain a parallel corpus. All the resulting text data was post-processed so as to match the speech normalization described above.

We split the call transcription corpus in two parts. Each part consists of about 7 minutes of speech. Part 1 (*calls1*), containing 176 sentences from a total of 17 calls, is used as a development set for tuning our ASR model. Part 2 (*calls2*) consists of 188 sentences from 18 calls and is used for evaluation of both speech recognition and machine translation.

how do you pronounce that	how far is it
i don't have a boyfriend	be careful
how come you don't understand	can i say anything else
i'd like to go home	do you have any meatless dishes
it hurts here	i hope not

Figure 2.2: A sample of English sentences from the reference corpora (normalized).

We also collected another speech corpus designed for testing the recognition of Czech names. This corpus is described in Section 3.3.

We made all of the speech corpora available [9] in the LINDAT/CLARIN digital library.

2.2.3 Reference corpora

We assembled data from two different sources that we consider to represent the travel domain well:

- Part 1 of the call transcription corpus (*calls1*);
- short sentences collected from Czech–English and English-only web phrasebooks (Local Lingo, Omniglot, SpeakLanguages).[7]

By manually editing the data to eliminate errors and nonsensical sentences, we obtained the following 'reference corpora':

- *lmppl-ref*: English 'phrasebook sentences',
- *mixlm-tuning*: cleaned English transcriptions from *calls1*,
- *mert-tuning*: cleaned parallel data from both sources (*calls1* & phrasebooks).

Sample English sentences from these corpora are given in Fig. 2.2.

2.2.4 Common Crawl

Our selection from the Common Crawl corpus is the main source of in-domain data for the ASR model. While the Common Crawl itself is domain-independent and of mixed quality, it is also massive (hundreds of terabytes of text), and we can therefore hope that it contains a sufficient number of high-quality in-domain sentences that we can find and use for training our language model. We use a considerably less impressive version of the corpus from WMT 2016,[8] consisting of no more than 103 GB of compressed English text data.

Since the corpus is not fully sentence-segmented, we first use the TrTok tokenizer [18] (with its *CzEng* tokenization model for English) to segment it. Next,

[7] `http://www.locallingo.com/czech/phrases/`, `http://www.omniglot.com/language/phrases/czech.php`, `http://www.speaklanguages.com/english/phrases/`.

[8] `http://www.statmt.org/wmt16/translation-task.html#download`

we filter it using a simplified version of the technique described by Gao et al. [11], with some additional constraints. A similar approach is used by Jurčíček [13] for training a domain-specific dialog system.

The most important criterion used for the selection of sentences is their *perplexity* according to a language model trained on a small in-domain corpus. For a test sentence $\mathbf{w}$, perplexity is computed as the inverse probability normalized by the number of words:

$$\mathrm{PP}(\mathbf{w}) = p(\mathbf{w})^{-\frac{1}{|\mathbf{w}|}}. \tag{2.1}$$

The higher the perplexity, the less predictable a sample is according to the model. Therefore, by selecting sentences with low perplexities, we should obtain a corpus which is similar to the training data.

First, we train a small language model using our English phrasebook sentence collection (*lmppl-ref*). Next, we normalize all sentences from the Common Crawl and calculate the perplexity of each of these sentences using this reference LM. We then select all sentences satisfying the following conditions:

- the sentence is at least 3 and less than 35 words long,
- less than 40 % of these words are out-of-vocabulary words (not present in the LM),
- the computed perplexity is less than a certain value.

These thresholds are chosen arbitrarily to produce a reasonable number of sentences that are fairly similar to the phrasebook corpus. A sample from the *cc-select-80* corpus (obtained by setting the perplexity threshold to 80) is shown in Fig. 2.3. We can see that the selected sentences do not represent the travel domain very well, but they are close in register to the reference corpus.

2.2.5 Wikipedia

We find it hard to extract useful data from Wikipedia due to its rich and inconsistent formatting. Moreover, it covers a vast amount of specialized topics in a very technical way, and therefore, in its entirety, it is probably not a good basis for a language model for our purposes. However, we include a selection of Wikipedia articles to contribute some Czech Republic-related *n*-grams.

We use an HTML dump of the English Wikipedia from May 2015 from the Kiwix project.[9] Our corpus consists of the text of the 32,294 ‘neighbors’ of the article *Czech Republic* (i.e. articles that either link to or are linked by Czech Republic, plus Czech Republic itself). Simple pre-processing is used: we remove footnote references from the document (these are labelled with a special HTML class), then concatenate the text contents of all paragraphs (`p` elements) and perform sentence segmentation using TrTok as in the previous section.

[9] `http://download.kiwix.org/zim/wikipedia/`

"But we've also heard – and seen, with Jim's game-play, that MMORPGs like World of Warcraft can be more engaging and distracting than other games, sucking away hours and hours in seemingly endless online quests. The Flat Slat Sleigh changing table can also double as a storage station for a bathroom or laundry room. Juni 2011 Filed under: Technik - chris @ 22:36 30 mile (s) from Longmont 334. I can't help but think it must be a beautiful life down there. 135-1009 The Village Stavrohori This is one of the bigger villages of the Sitia District, listed in 1583 by the Venetians as Stavrodoxari with 227 Cats and Things Xbox Video Games. NOTES:	It's surprising what you don't know. Thank you so much. I am impressed by the details that you have on this blog. Let's hope so. I missed this one. "I don't know if I'm better. Click on the icon on the right to see the full pool. He is that person to me. You have been busy. I love to go out and have a good time. It is the shortest of the books of the Codex Calixtinus. I like the trumpet.

Figure 2.3: A random sample from the Common Crawl, without normalization. Left: unfiltered. Right: selected using the reference LM, with the perplexity limit equal to 80 (more sentences are shown in order to make use of the space).

2.3 ASR training

2.3.1 Language model

On each of the mentioned corpora (parts of CzEng, Common Crawl and Wikipedia), we train a separate 5-gram language model. We remove from the models all *n*-grams that contain words which are not in our pronunciation lexicon (see Section 2.3.2). These are words that cannot be recognized because their pronunciation is unknown. In our case, this reduces the number of unigrams in each of the models by about 85–90 % and the total number of *n*-grams by about 15 %.[10]

Subsequently, we combine these models using interpolation [33]: we assign each of the models a weight, take the union of their *n*-grams, assign each *n*-gram the weighted average of the probabilities from those models, and then renormalize the new model. The interpolation weights are tuned to minimize perplexity on the reference corpus. We use the interpolation script[11] from Moses, which uses tools from SRILM to compute the optimal weights and run the interpolation.

We need to further reduce the model to make it practically usable for speech recognition in terms of speed and memory requirements. We achieve this using SRILM's entropy-based pruning [32]: we remove from the LM all *n*-grams such that their removal leads to an increase in perplexity (on the training set) by less than a certain threshold.

[10] The percentage of removed unigrams seems very high. By consulting the MT phrase table, we found that about 80 % of these words were unknown to the translation model, and therefore more or less safe to remove. The remaining 20 % seemed to be a mixture of relatively rare, misspelled or incomplete English words, and words from other languages.

[11] `https://github.com/moses-smt/mosesdecoder/blob/master/scripts/ems/support/interpolate-lm.perl`

consonants				vowels/diphthongs			
Czech	ex.	English	ex.	Czech	ex.	English	ex.
t͡s	**c**ár	t s	**ts**unami	o	**o**ko	ɔ	**ough**t
c	**ť**apka	t͡ʃ	**ch**eese	a͜u	**au**to	a͜ʊ	c**ow**
ɟ	**ď**ábel	d͡ʒ	**j**eans	e͜u	**eu**ro	æ u	—
ɲ	**ň**adra	n	**n**eed	o͜u	**ou**ško	o͜ʊ	**oa**t
r	**r**ád	ɹ	**r**ead				
r̝	**ř**ád	ʒ	vi**si**on				
r̝̊	k**ř**áp	ʃ	**sh**e				

Table 2.2: The mapping of the Czech-only phones to English phones. Each phone or group of phones is provided with an example in the respective language.

2.3.2 Pronunciation lexicon

We use the popular Carnegie Mellon University Pronouncing Dictionary[12] (also known as CMUdict), containing over 134 k English words.

In one of our experiments, we attempt to enable the recognition of certain Czech named entities by adding their phonetic representations to the lexicon. We proceed as follows:

1. Download the database of Czech addresses from the website of the Ministry of the Interior[13] and extract the names of all municipalities, quarters and streets.

2. Using a phonetic transcription script[14] from Alex, generate the Czech pronunciations of all words appearing in the data obtained in the previous step (except for a few hand-picked common English words like *a* and *do*).

3. Express the Czech pronunciations using the phone set used by CMUdict and add them to the lexicon.

The third step deserves further explanation. The Czech and English phone sets are very different, and although some phones are shared across the two languages, a number of Czech phones don't have English equivalents.[15] For these phones, we attempted to find the closest match in terms of manner and place of articulation, as shown in Table 2.2. Note that the 'phones' produced by the script used in step 2 would be more accurately described as *phonemes*, with the exception of the allophones [r̝] and [r̝̊] (voiced and voiceless /ř/), which are indeed treated as two separate units. We left this behaviour unchanged. For an account of Czech phonetics and phonology, see e.g. [12].

[12] `http://www.speech.cs.cmu.edu/cgi-bin/cmudict`

[13] `http://www.mvcr.cz/clanek/databaze-adres-v-cr-a-ciselniky-uzemnich-celku.aspx`

[14] `https://github.com/UFAL-DSG/alex/blob/master/alex/tools/kaldi/local/phonetic_transcription_cs.pl`

[15] We consider two phones equivalent if they are represented by the same symbol in the International Phonetic Alphabet (IPA).

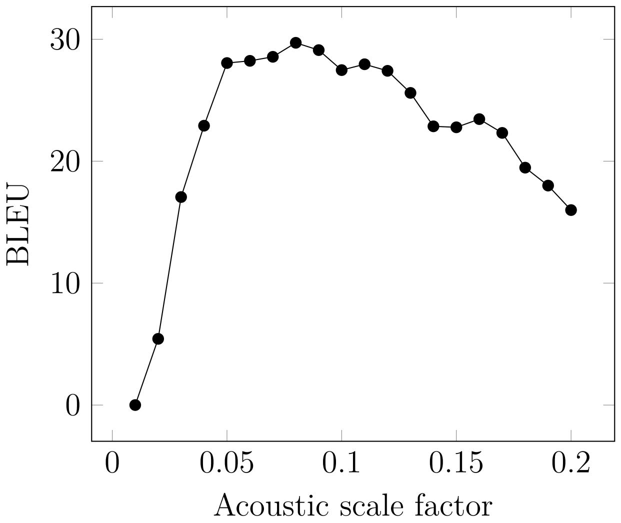


Figure 2.4: The BLEU score of the STEM-ALI system on the development set for different values of the acoustic scale factor. (See Section 3.1 for more information about this system and how it is evaluated.)

2.3.3 Final model

We use an existing acoustic model trained on approximately 2,000 hours of telephone speech from various sources, namely USC-SFI MALACH [30], SWITCHBOARD [8] and LibriSpeech [25]. The UFC-SFI data, consisting of interviews with survivors and other witnesses of the Holocaust, is characterized by natural, spontaneous speech with a variety of non-native accents, which makes it particularly suitable for our purpose.

The recognizer has several parameters which need to be set, most importantly the acoustic scale factor, i.e. the weight of the acoustic model relative to that of the language model. We tune the acoustic scale factor by using linear search to find the value that maximizes the BLEU score of the entire system on the development set (*calls1*). The effect of this parameter on the BLEU score is illustrated in Fig. 2.4.

The other parameters – *beam*, *lattice-beam* and *max-active-states* – control the speed of the recognizer while also affecting the recognition accuracy. We set these parameters according to Plátek and Jurčíček [28] and then tweaked them to reduce the latency on the development set without decreasing the BLEU score significantly. Our settings are: *beam* = 13, *lattice-beam* = 3, *max-active* = 1500.

2.4 MT training

We train a translation model on the *czeng16* parallel corpus and a separate 5-gram language model on each of the Czech corpora, as shown in Table 2.1. The TM and the LMs are all used as separate features in a log-linear model, with weights optimized on the *mert-tuning* corpus (phrasebooks and manual call transcriptions). For the translation model, we choose between two variants of alignment:

a) computed on raw tokens.

b) computed on word ‘stems’, obtained by taking the first four characters of each token. When pre-processing the corpus for stemming, we skip step 3 of the normalization procedure, so that we use raw word forms instead of the non-speech event symbols.

We shall not go into greater detail about the MT training since we use the standard Eman pipeline, mostly with default settings.

3. Experiments

3.1 Baseline and speech normalization

Fig. 3.1 shows the individual steps of the BASELINE experiment and their mutual dependencies. For the baseline, we limited ourselves to the most basic pre-processing. In the corpora used for the ASR language model (top right of the figure), we used the default pre-processing from Alex, which merely lowercases the text and replaces all non-letters with spaces. For machine translation (left side of the figure), we used a tokenizer bundled with Eman.

Instead of the perplexity-based data selection from the Common Crawl corpus, we simply selected 46 M random sentences. We pruned the final ASR language model using a perplexity threshold of 10^{-8}.

We ran two additional experiments: The setup of SP-NORM is identical to that of BASELINE, but the pre-processing of all corpora is replaced with speech normalization as described in Section 2.2.1. In STEM-ALI, we additionally used the alignment on stems from Section 2.4.

3.1.1 Evaluation

We performed an end-to-end evaluation of each system by first decoding the sentences in Part 2 of our call recording corpus (*calls2*) using the speech recognition model, then translating the recognized sentences using the MT model, and finally computing the BLEU score of the result against the human translations of the original sentences.

We also tested the performance of both components of the system separately. To evaluate the speech recognition model, we measured the word error rate (WER) of the decoded sentences against the manual transcriptions of the recordings. Similarly, we evaluated the MT model in isolation by using it to translate the manual transcriptions and then computing the BLEU score with their human translations as a reference.

For evaluation, we removed any punctuation that the MT component might have produced. Because the reference translations do not contain punctuation either (as discussed in Section 2.2.2), we are effectively measuring the BLEU score only on words.

The results are shown in Table 3.1, including also several other systems for comparison (these are discussed below and in Section 3.2). Apparently, the SP-NORM and STEM-ALI systems present a great improvement over the BASELINE (the end-to-end BLEU score increases from 22.99 to 27.93 for SP-NORM and 31.88 for STEM-ALI). We suspect, however, that the variation in WER in these cases (27.61 → 25.25 → 27.19) is largely owing to the fact that the ASR models were tuned for use with different MT models. This is clearly the case of the last two of these three systems (SP-NORM and STEM-ALI), where the same ASR model is used, except for the value of the acoustic scale factor.

In order to assess to what extent translation quality can be improved by adding punctuation to the input, as opposed to removing it from the training data, we also included a version of the baseline system that uses ‘oracle punctuation’. This

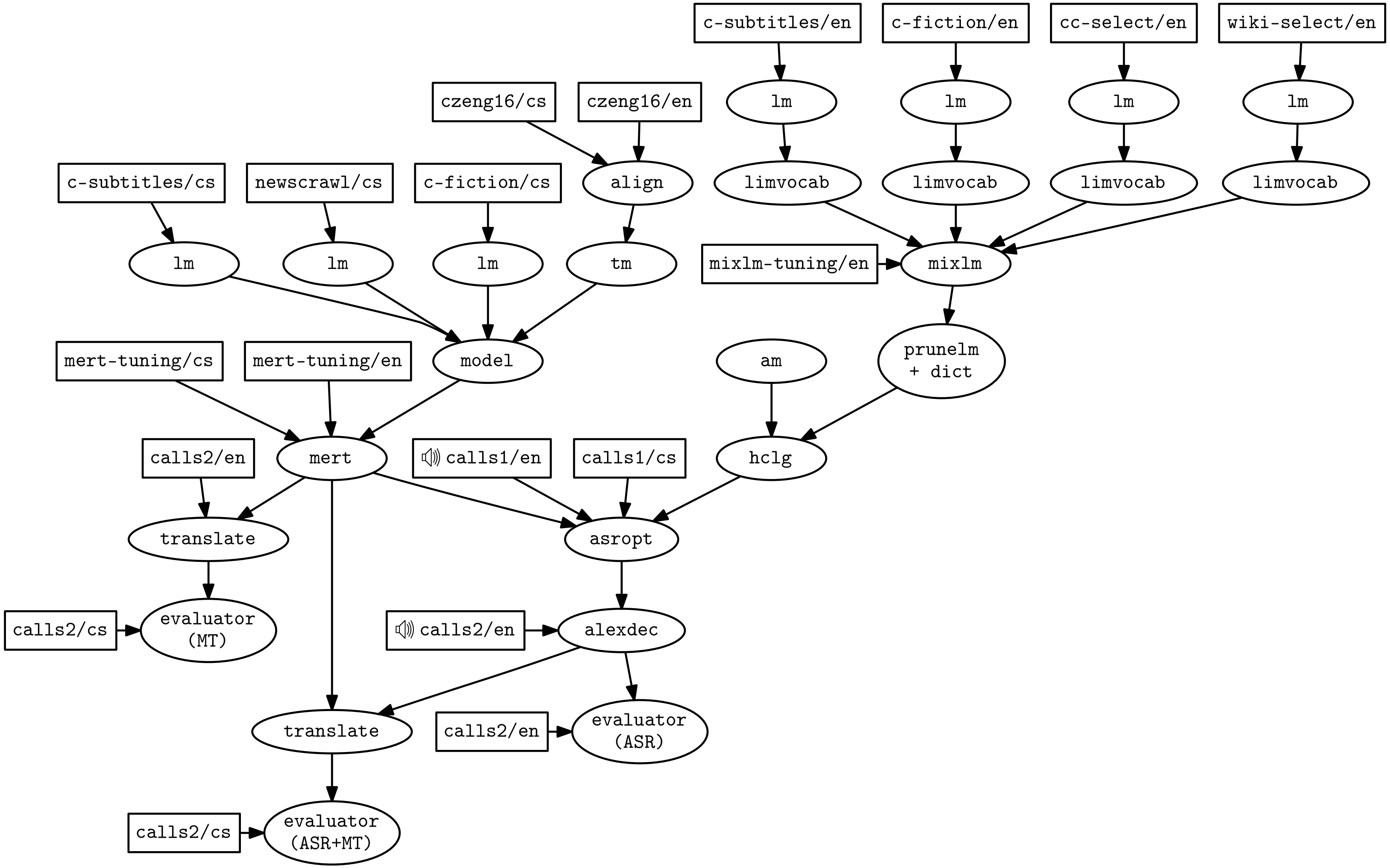


Figure 3.1: The structure of the baseline experiment (simplified). Each node corresponds to one Eman step, rectangles denote corpora. See Appendix A.1 for a brief description of each step type.

System	% BLEU (ASR+MT)	% WER (ASR)	% BLEU (MT)
BASELINE	22.99	27.61	44.22
+ oracle punctuation*	25.62	—	47.76
SP-NORM	27.93	25.25	45.25
STEM-ALI	31.88	27.19	**48.50**
SELECT-80B	28.67	23.94	—
SELECT-100A	**34.64**	22.92	—
GASR-CHIMERA	31.75	13.57	31.78

*Not available in practice.

Table 3.1: The performance of our systems on the test set (*calls2*). The MT-only BLEU scores for the systems from Section 3.2 are not shown since the MT component is the same as in STEM-ALI. The performance of the prototype (Google ASR + Chimera) is included for comparison.

is not a real system, but a hypothetical one where the punctuation of the input sentence is correctly guessed by an oracle before the sentence is translated. We achieved this by adding the punctuation by hand to the manual transcriptions and the output of ASR before running the machine translation.

We can see that using oracle punctuation in the BASELINE system helps more (BLEU 47.76) than simply ignoring punctuation (as SP-NORM does, reaching only a BLEU of 45.25), but only when the input is a human transcription. When the output of ASR is being translated, the result is opposite (25.62 vs. 27.93 BLEU). This suggests that with a better ASR model, using automatic punctuation prediction on its output before passing it to an MT model trained on punctuated data could yield better results.

3.2 Data selection

In the next experiment, we focused on the speech recognition model, testing different thresholds (denoted σ in the following) for the perplexity-based selection of data from the Common Crawl, as well as different pruning thresholds (denoted τ) for the ASR language model. We used the STEM-ALI setup as a basis for these experiments.

The settings we examine here are $\sigma = 60, 80, 100$ and $\tau = 10^{-7}, 10^{-8}, 10^{-9}$; for less constraining values of either parameter, the speech recognition model becomes highly computationally intensive, requiring more than 100 GB of RAM for building and more than 30 GB for decoding.

3.2.1 Automatic evaluation

We only evaluate the ASR and ASR+MT performance because the MT model is the same throughout the experiment.

We found that if we replace the 46 M random sentences form the Common Crawl with the same number of sentences selected based on perplexity (with $\sigma = 80$), the accuracy of speech recognition increases, but the overall BLEU

System	% ≥ other	% acceptable
BASELINE	41.49	28.19
SELECT-100A	**61.70**	**37.77**

Table 3.2: The results of manual evaluation of the best system against the baseline.

score decreases. This is shown in Table 3.1 (STEM-ALI → SELECT-80B: decrease in both WER and BLEU). However, the two models are difficult to compare since they are not of the same size: as we have seen in Fig. 2.3, the random selection contains much longer sentences, and therefore the resulting language model has a much higher number of n-grams.

If, for a given choice of σ, we decrease the pruning threshold τ, thereby increasing the size of the training data, both ASR and ASR+MT tend to improve. This is apparent from Fig. 3.2 and Fig. 3.3, which show the results depending on the total number of n-grams in the pruned LM.

No general conclusion can be drawn regarding the effect of σ; quite possibly, our test set is too small, leading to excessive noise in the results. However, with the least aggressive pruning ($\tau = 10^{-9}$), increasing σ seems to help in terms of BLEU ($\sigma = 60 \rightarrow 80 \rightarrow 100$ led to a gradual increase in BLEU by 4.7 points). Indeed, the best result overall (BLEU-wise), shown in Table 3.1 as SELECT-100A, was achieved using the settings $\tau = 10^{-9}$, $\sigma = 100$.

3.2.2 Human evaluation

We also performed a manual evaluation of the best system, SELECT-100A, against the BASELINE. We compared the two translations of each sentence in the test set and we recorded for each system the number of times it was judged *better than or equal* to the other system. (Therefore, if the outputs of both systems were equally good or equally bad, they would both get a +1.) We were also interested in how often each system produced an 'acceptable' translation, i.e. one that was understandable and retained the original meaning. The results are shown in Table 3.2.

3.3 Czech names recognition

In order to evaluate the ability of our speech recognition models to recognize Czech names, we needed a speech corpus containing a sufficient number of them. Because the test set used for all other experiments doesn't satisfy this requirement, we resorted to an artificial one. We constructed a set of 54 short English sentences, each containing the Czech names of one or more places in the Czech Republic, and asked our five colleagues (foreigners studying in Prague) to record them by means of Alex Translate. We filtered the recordings to remove defects such as stuttering or incomplete sentences, but we didn't exclude any mispronunciations of the Czech expressions. We performed evaluation on the resulting set of 256 sentences (*cstest*) by calculating the WER and also the percentage of correctly recognized place names.

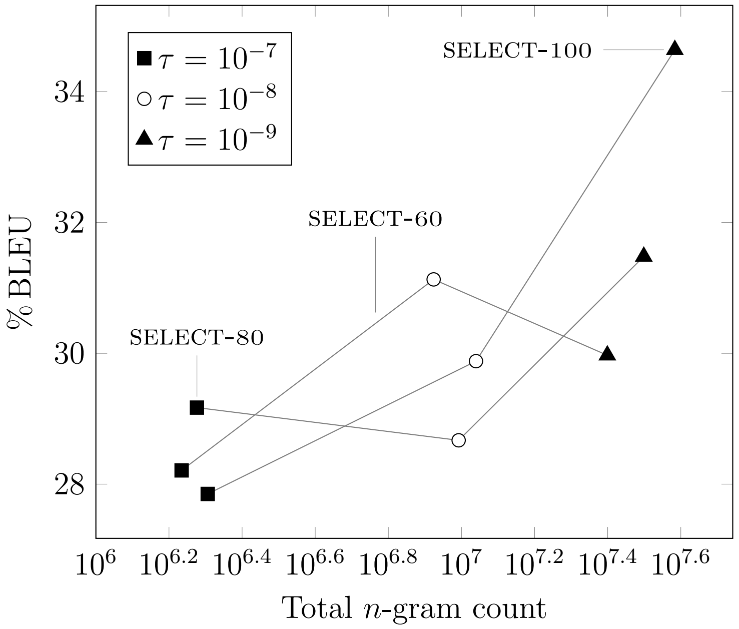


Figure 3.2: The end-to-end BLEU scores for different settings; higher pruning thresholds tend to, but do not always, lead to better (higher) BLEU scores.

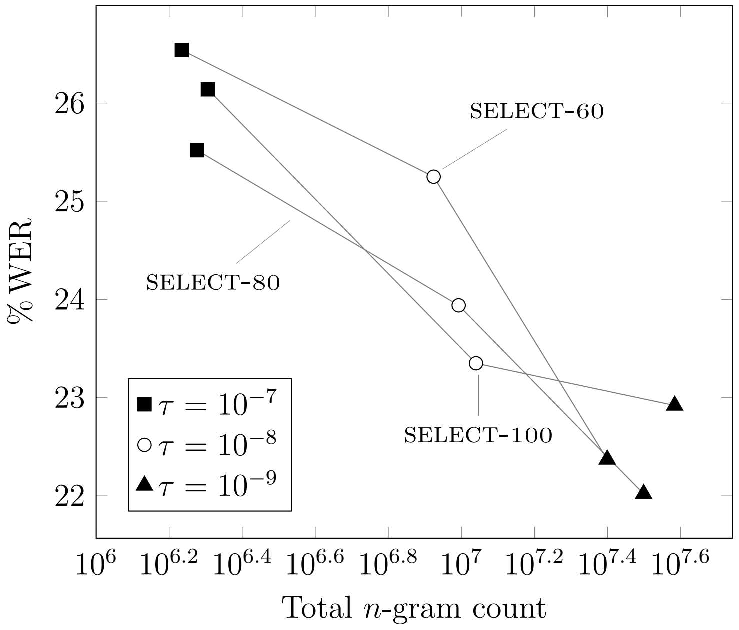


Figure 3.3: The WER scores for ASR for different settings; higher pruning thresholds always lead to better (lower) WER scores.

System	% WER		% correct
	calls2	*cstest*	*cstest*
SELECT-80A	22.02	64.79	0.00
CS-NAMES	21.92	**61.85**	**6.39**

Table 3.3: The performance of two different speech recognition models on our 'regular' test set and on an artificial set of sentences containing Czech names.

We evaluate two systems: SELECT-80A ($\sigma = 80$, $\tau = 10^{-9}$) and CS-NAMES. The latter system was obtained by re-building the former with the extended pronunciation lexicon described in Section 2.3.2. We chose $\sigma = 80$ instead of 100 to reduce the memory cost since the bigger lexicon leads to a bigger, more demanding model. Moreover, we have seen in Fig. 3.3 that SELECT-80A performed slightly better than SELECT-100A in terms of WER.

In the first column of Table 3.3, we can verify that the extended lexicon doesn't cause a significant increase in WER on the *calls2* corpus (in fact, it leads to a decrease by 0.1). The more interesting part is the evaluation on the *cstest* set; while the extended lexicon presents a noticeable improvement (-2.94 WER), the performance is obviously still unsatisfactory: only 6.39 % of the place names were recognized correctly and the WER of 61.85 indicates that the overall word accuracy was lower than 40 %.

By inspecting the output of CS-NAMES, we found that the model seems to favour certain place names: out of the 17 correctly recognized names, *Dejvická* was repeated three times and *České Budějovice* and a few others twice. While most (96 %) of the Czech vocabulary necessary to recognize all of the names is indeed present in the model, it could be the case that the n-grams containing the unrecognized names are not present in the LM with sufficient probabilities, for example because they were removed during pruning. If this were true, it would help increase the probabilities of such n-grams in the LM, e.g. using a specialized training corpus.

It is also possible that our phone mapping is too distorting, or that the unrecognized names consist of phone sequences that are uncommon in English, and therefore disqualified by the acoustic model. Better performance could be achieved by using a one-to-many phone mapping instead of a one-to-one mapping, allowing multiple pronunciations per word, or by considering sequences of Czech phones instead of single phones. Nevertheless, in order to reach high recognition accuracy for Czech words, the AM would probably need to be re-trained, perhaps using the approach mentioned in Section 1.1.5.

3.4 Speed

For real-time speech translation, which is the intended application of our system, low response time of the ASR and MT models is essential. We measured the *latency* of the models for each of the sentences in the test set. For on-line speech recognition, latency is the delay between the end of the utterance (the moment the user has finished speaking) and the availability of the recognition result; for

System	ASR+MT		ASR		MT	
	50 %	95 %	50 %	95 %	50 %	95 %
SELECT-100A	0.67	4.87	0.07	0.33	0.58	4.78
GASR-CHIMERA	1.35	2.68	0.80	1.65	0.40	1.52

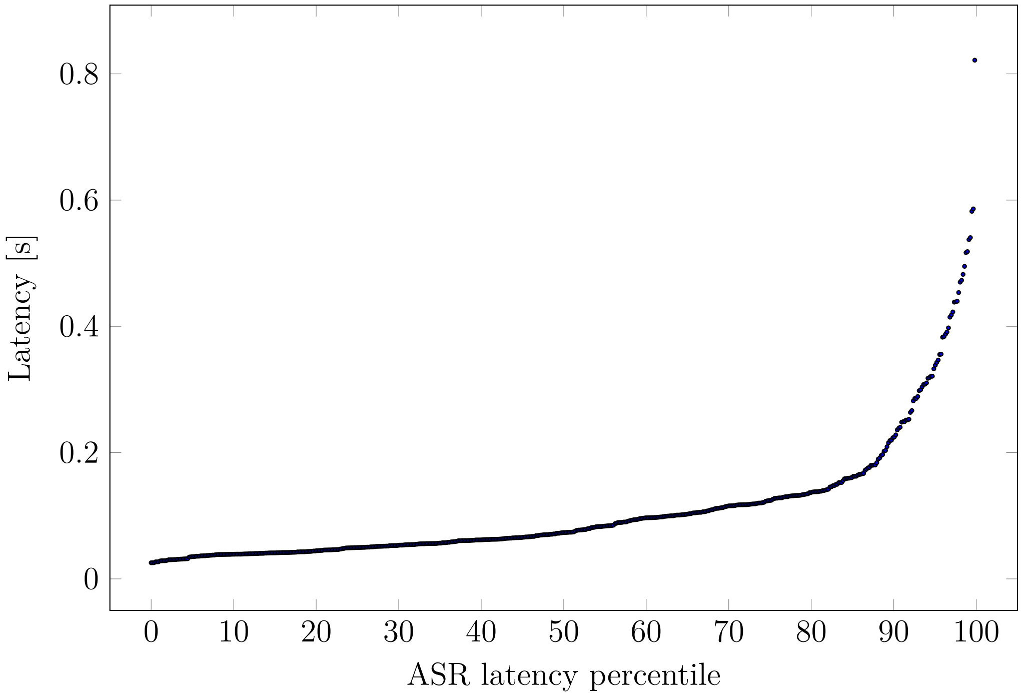
Table 3.4: Median and 95th percentile latencies in seconds, measured on the test set.

MT, it is simply the time consumed by translation. The sum of these two latencies is the total latency of the system, i.e. the total amount of time the user has to wait before the translation is available.

In Table 3.4, we report the median latency and the 95th percentile latency of our SELECT-100A system and its components. The data was collected in three consecutive runs of each component on the same machine (hence, each test sentence was recognized and translated three times). For comparison, the same statistics are reported for GASR-CHIMERA. Of course, because both components of the latter system are web-based, their latencies are expected to vary depending on the Internet connection quality.

While the latency of our ASR model is a noticeable improvement over Google ASR (probably because we are running the recognizer locally, eliminating the overhead of the web API), our MT latency is very high. This is even more evident from Fig. 3.4: the latency of ASR stays in the order of fractions of a second, but translation takes more than 10 s in more than 3 % of the cases. This renders the system virtually unusable for real-time translation.

We should stress, however, that we did not optimize the MT component for speed at all, and that there are many possibilities in this regard. The Moses decoder has options that allow to limit the search space (i.e. the set of possible translations that are considered) [15], therefore speeding up the decoding. Also, the model itself can be filtered, for example, making it more computationally efficient. Unfortunately, these adjustments usually have a negative effect on the output of the system. It is therefore necessary to find a balance between speed and translation quality.



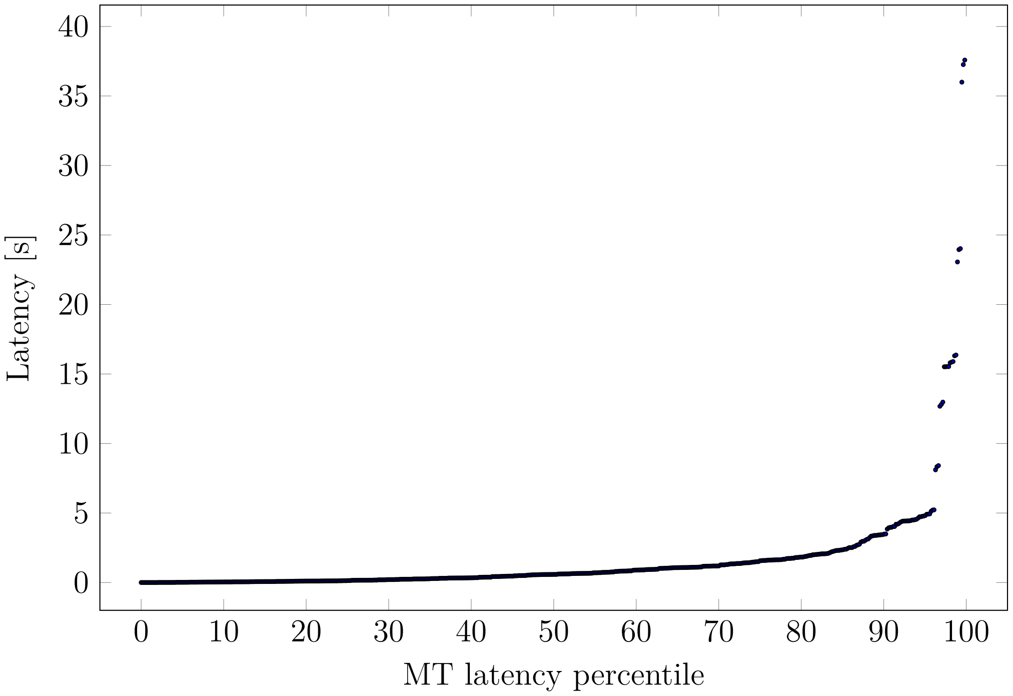


Figure 3.4: Percentile graphs of the ASR and MT latency of the SELECT-100A system on the individual sentences in the test set (each included three times).

Conclusion

We presented Alex Translate, our English-to-Czech spoken language translation system, and proposed improvements to its ASR and MT components. We conducted a series of experiments in which we evaluated these improvements, using both automatic metrics and human judgement. We conclude that our improvements were generally successful with regard to translation quality, but the system is not yet ready for deployment within Alex Translate due to high response times of the MT model.

Future work

The next step should naturally be to optimize the MT model for real-time translation.

It would also be desirable to investigate why our approach to foreign words recognition was so unsuccessful, and attempt to improve it. As discussed earlier, improvement could be achieved either by using a specialized corpus or training a custom acoustic model.

Another possible direction of future research is punctuation prediction. This is a widely used technique in SLT and we have suggested that it might bring a small improvement in our case as well.

Lastly, our system could be improved by introducing a more complex interface between speech recognition and machine translation. Specifically, the use of word lattices has proven useful in SLT. The recognizer we use does produce word lattices, and the Moses decoder is capable of decoding them efficiently, so this should be fairly straightforward to implement in our setup.

Acknowledgement

This work has been supported by the LINDAT/CLARIN project of the Ministry of Education, Youth and Sports of the Czech Republic (project LM2015071). The research was also partially funded by the Ministry of Education, Youth and Sports of the Czech Republic under the grant agreement LK11221.

A. Attachment: Playground

The attachment to this thesis contains our copy of the UFAL SMT Playground, which we used to run our experiments. Since the playground has many dependencies and is not easily portable between different machines, we include it merely for illustrative purposes. We also had to remove most of the files in order to reduce the size of the playground from the original 155 GB to about 3 GB. Nonetheless, we preserved the essential training data (except for the CzEng corpus and its parts) and the descriptions of all the steps, so a skilled Eman user should be able to replicate our results with some effort.

Below, we give a concise description of the playground and our additions to it. Our contributions are recorded in the `slt` branch of the Git repository.[1]

A.1 Steps, seeds and tracebacks

The `playground` directory contains a number of subdirectories with names of the form `s.`*`steptype.hash.date-time`*; each represents one step of an experiment and its name encodes the *type* of the step, a unique hash and the time it was created. For example, the directory `s.corpus.04f0a0a5.20160419-2023` represents the step containing the *newscrawl* corpus.

Each step was generated from a *seed*, which is a script that creates steps of a particular type. Each seed defines a set of variables, whose values are specified at the creation of a new step. Based on these variables, the seed prepares the contents of the step directory, most importantly the `eman.command` file, which contains the actual code that produces some useful data.

The most important seeds (step types) used in our experiments are:

`corpus` Registers a new corpus.

`lm` Trains a language model on a given corpus.

`mixlm` Interpolates a given set of language models, with weights tuned on a development set.

`align` Produces word alignment for a given parallel corpus.

`tm` Trains a translation model on a given aligned parallel corpus.

`model` Prepares a model for Moses from a set of `tm`s and `lm`s.

`mert` Tunes a `model`'s weights using MERT.

`translate` Translates a corpus using a given `mert` step.

`evaluator` Calculates the BLEU or WER score of the output of MT or ASR.

`hclg` Builds a Kaldi speech recognition model from a given acoustic model, language model and pronunciation lexicon.

[1] `https://redmine.ms.mff.cuni.cz/projects/ufal-smt-playground/repository/show?rev=slt`

alexdec Runs speech recognition on a speech corpus using a given `hclg` model or Alex configuration file.

hack This is a meta-seed that allows to create a step with arbitrary code without having to write a new seed, and then easily replicate it. Note that in Fig. 3.1, we display steps of type `hack` under more descriptive labels.

We created the last 3 seeds for the purposes of our experiments; the rest are a standard part of the SMT Playground.

The *traceback* of a given step is the tree of the predecessors of that step (i.e. all steps it transitively depends on). With Eman installed, one can run the command[2]

```
eman tb SPEC
```

to obtain the traceback of the step specified by *`SPEC`*. The option `--vars` can be added, causing Eman to also display the values of all variables of each step in the traceback. *`SPEC`* can be the name of a step or anything that uniquely identifies a step; for example, in our particular setup, an `evaluator` can be referenced using the computed score. Therefore, we can use

```
eman tb --vars 34.64
```

to view the complete specification of the experiment that achieved the BLEU score 34.64.

A.2 Scripts

Besides step directories, the playground also contains a collection of tools that are used by the steps. Here, we provide the user documentation for two scripts, `normalize-speech.pl` and `phonetic_transcription_cs_to_en.pl`, which we created as a part of this thesis. Both scripts are located in `playground/tools`.

A.2.1 normalize-speech.pl

The script applies speech normalization to its standard input and writes the result to its standard output. The description in Section 2.2.1 corresponds to the following usage (with the language specified by `--lang`):

```
normalize-speech.pl --lc --numbers=convert --nohyp --nonspeech
```

Options

`--column=N`
Normalize only the *`N`*-th tab-delimited column.

`--keep-nonspeech`
Preserve non-speech event symbols.

[2] The command has to be executed inside the `playground` directory, which has to be writable so that Eman can index it.

`--keepempty`
Do not omit empty lines.

`--lang=en|cs`
Use the given language. Default is `en`.

`--lc`
Lowercase the text (uppercase by default).

`--nohyphens`
Do not leave hyphens in the input.

`--nonspeech`
Replace words like *uh* and *er* (depending on the language) with the hesitation symbol `_EHM_HMM_`.

`--numbers=keep|convert|remove`
Specify what to do with numbers. Default is `remove`.

`--punct`
Keep punctuation.

`--transcription`
Indicates that the input is the result of manual transcription. Implies the `--nonspeech` option unless negated by `--nononspeech`.

A.2.2 phonetic_transcription_cs_to_en.pl

This script expects the output of `phonetic_transcription_cs.pl` on the standard input. It replaces the Czech phones by the English phones used in the CMU dictionary (as shown in Table 2.2) and writes the result to the standard output. Each line of the input is expected to contain a word and a sequence of phones, all separated by whitespace.

Bibliography

[1] Ondřej Bojar and Aleš Tamchyna. The design of Eman, an experiment manager. *The Prague Bulletin of Mathematical Linguistics*, 99:39–56, 2013.

[2] Ondřej Bojar and Aleš Tamchyna. CUNI in WMT15: Chimera strikes again. In *Proceedings of the 10th Workshop on Statistical Machine Translation*, pages 79–83, Stroudsburg, PA, USA, 2015. Association for Computational Linguistics.

[3] Ondřej Bojar, Zdeněk Žabokrtský, Ondřej Dušek, Petra Galuščáková, Martin Majliš, David Mareček, Jiří Maršík, Michal Novák, Martin Popel, and Aleš Tamchyna. The joy of parallelism with CzEng 1.0. In *Proceedings of LREC2012*, Istanbul, Turkey, May 2012. ELRA, European Language Resources Association. In print.

[4] Christian Buck, Kenneth Heafield, and Bas van Ooyen. N-gram counts and language models from the Common Crawl. In *Proceedings of the Language Resources and Evaluation Conference*, Reykjavk, Icelandik, Iceland, May 2014.

[5] Francisco Casacuberta, Hermann Ney, Franz Josef Och, Enrique Vidal, Juan Miguel Vilar, Sergio Barrachina, Ismael Garcıa-Varea, David Llorens, César Martınez, Sirko Molau, et al. Some approaches to statistical and finite-state speech-to-speech translation. *Computer Speech & Language*, 18 (1):25–47, 2004.

[6] Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. Report on the 11th IWSLT evaluation campaign, IWSLT 2014. In *IWSLT*, pages 2–17, 2014.

[7] Stanley F Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13(4):359–393, 1999.

[8] Christopher Cieri, David Graff, Owen Kimball, Dave Miller, and Kevin Walker. Fisher English training Part 2, Speech, 2005. URL `https://catalog.ldc.upenn.edu/LDC2005S13`. Linguistic Data Consortium.

[9] Ondřej Cífka and Ondřej Bojar. A small dataset for English-to-Czech speech translation in the travel domain, 2016. URL `http://hdl.handle.net/11234/1-1735`. LINDAT/CLARIN digital library at Institute of Formal and Applied Linguistics, Charles University in Prague.

[10] Ondřej Dušek, Ondřej Plátek, Lukáš Žilka, and Filip Jurčíček. Alex: Bootstrapping a spoken dialogue system for a new domain by real users. In *15th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, page 79, 2014.

[11] Jianfeng Gao, Joshua Goodman, Mingjing Li, and Kai-Fu Lee. Toward a unified approach to statistical language modeling for Chinese. *ACM Transactions on Asian Language Information Processing (TALIP)*, 1(1):3–33, 2002.

[12] Miroslav Grepl, Petr Karlík, Marek Nekule, Zdenka Rusínová, and Petr Karlík. *Příruční mluvnice češtiny*. Nakladatelství Lidové noviny, Praha, 2012.

[13] Filip Jurčíček. Approach to bootstraping the domain specific language models. `http://alex.readthedocs.org/en/master/_man_rst/alex.applications.LANGUAGE_MODELLING.html`, 2013. Online; accessed 8 April 2016.

[14] Filip Jurčíček, Ondřej Dušek, Ondřej Plátek, and Lukáš Žilka. Alex: A statistical dialogue systems framework. In *Text, Speech and Dialogue*, pages 587–594. Springer, 2014.

[15] Philipp Koehn. Moses machine translation system user manual and code guide. `http://www.statmt.org/moses/manual/manual.pdf`, 2016. Online; accessed 13 June 2016.

[16] Philipp Koehn, Franz Josef Och, and Daniel Marcu. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 48–54. Association for Computational Linguistics, 2003.

[17] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 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, 2007.

[18] Jiří Maršík and Ondřej Bojar. Trtok: a fast and trainable tokenizer for natural languages. *The Prague Bulletin of Mathematical Linguistics*, 98: 75–85, 2012.

[19] Evgeny Matusov and Hermann Ney. Lattice-based ASR-MT interface for speech translation. *Audio, Speech, and Language Processing, IEEE Transactions on*, 19(4):721–732, 2011.

[20] Evgeny Matusov, Arne Mauser, and Hermann Ney. Automatic sentence segmentation and punctuation prediction for spoken language translation. In *IWSLT*, pages 158–165. Citeseer, 2006.

[21] Thipe Modipa and Marelie H Davel. Pronunciation modelling of foreign words for Sepedi ASR. 2010.

[22] Franz Josef Och. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 160–167. Association for Computational Linguistics, 2003.

[23] Franz Josef Och and Hermann Ney. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 295–302. Association for Computational Linguistics, 2002.

[24] Franz Josef Och and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003.

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

[26] Kishore Papineni, 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*, pages 311–318. Association for Computational Linguistics, 2002.

[27] Michael Paul, Marcello Federico, and Sebastian Stüker. Overview of the IWSLT 2010 evaluation campaign. In *IWSLT*, volume 10, pages 3–27, 2010.

[28] Ondrej Plátek and Filip Jurčíček. Free on-line speech recogniser based on Kaldi ASR toolkit producing word posterior lattices. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 108–112, 2014.

[29] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. The Kaldi speech recognition toolkit. In *IEEE 2011 workshop on automatic speech recognition and understanding*, number EPFL-CONF-192584. IEEE Signal Processing Society, 2011.

[30] Bhuvana Ramabhadran, Samuel Gustman, William Byrne, Jan Hajic, Douglas Oard, J. Scott Olsson, Michael Picheny, and Josef Psutka. USC-SFI MALACH interviews and transcripts English, 2012. URL `https://catalog.ldc.upenn.edu/LDC2012S05`. Linguistic Data Consortium.

[31] Georg Stemmer, Elmar Nöth, and Heinrich Niemann. Acoustic modeling of foreign words in a German speech recognition system. In *INTERSPEECH*, pages 2745–2748, 2001.

[32] Andreas Stolcke. Entropy-based pruning of backoff language models. In *DARPA Broadcast News Transcription and Understanding Workshop*, pages 270–274, 1998. Lansdowne, VA.

[33] Andreas Stolcke. SRILM – an extensible language modeling toolkit. In *Proceedings of the 7th International Conference on Spoken Language Processing (ICSLP 2002)*, pages 901–904, 2002.

[34] Keh-Yih Su, Ming-Wen Wu, and Jing-Shin Chang. A new quantitative quality measure for machine translation systems. In *Proceedings of the 14th conference on Computational linguistics-Volume 2*, pages 433–439. Association for Computational Linguistics, 1992.

[35] Lei Wang and Rong Tong. Pronunciation modeling of foreign words for Mandarin ASR by considering the effect of language transfer. In *INTERSPEECH*, pages 1443–1447, 2014.