PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 104 OCTOBER 2015 39–50

Sampling Phrase Tables for the Moses Statistical Machine Translation System

Ulrich Germann

University of Edinburgh

Abstract

The idea of virtual phrase tables for statistical machine translation (SMT) that construct phrase table entries on demand by sampling a fully indexed bitext was first proposed ten years ago by Callison-Burch et al. (2005). However, until recently (Germann, 2014) no working and practical implementation of this approach was available in the *Moses* SMT system.

We describe and evaluate this implementation in more detail. Sampling phrase tables are much faster to build and are competitive with conventional phrase tables in terms of translation quality and speed.

1. Introduction

Phrase-based statistical MT translates by concatenating phrase-level translations that are looked up in a dictionary called the *phrase table*. In this context, a phrase is any sequence of consecutive words, regardless of whether or not it is a phrase from a linguistic point of view. In addition to the translation options for each phrase, the table stores for each translation option a number of scores that are used by the translation engine (*decoder*) to rank translation hypotheses according to a statistical model.

In the *Moses* SMT system, the phrase table is traditionally pre-computed as shown in Fig. 1. First, all pairs of phrases up to an arbitrary length limit (usually between 5 and 7 words), and their corresponding translations are extracted from a word-aligned parallel corpus, using the word alignment links as a guide to establish translational correspondence between phrases. Phrase pairs are scored both in the forward and backward translation direction, i.e., $p\ (target\,|\,source)$ and $p\ (source\,|target)$, respectively. Computing these scores is traditinally done by sorting the lists on disk first to facili-

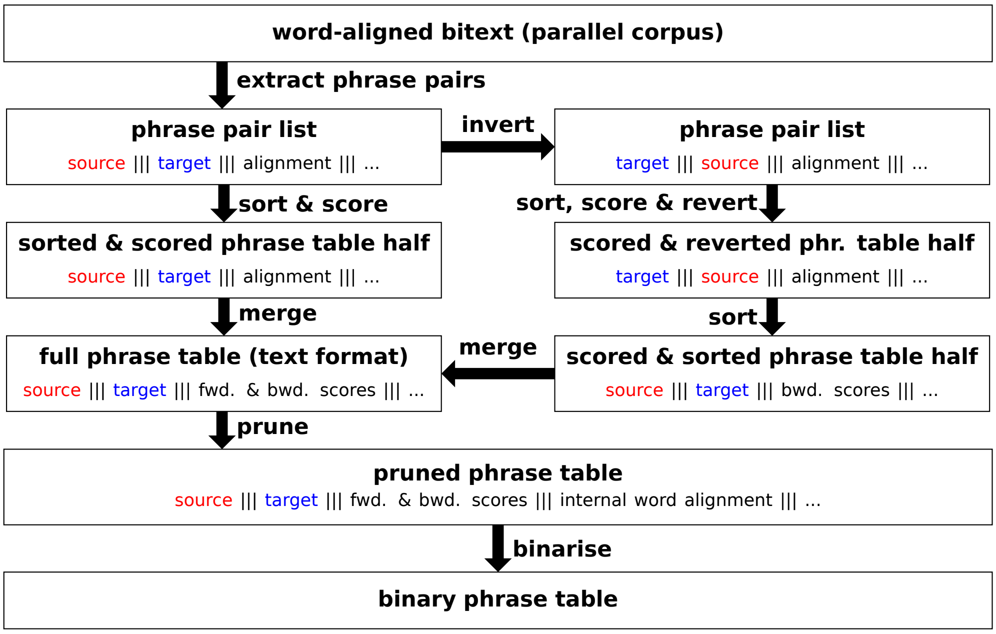
word-aligned bitext (parallel corpus)
extract phrase pairs
phrase pair list
source ||| target ||| alignment ||| ...
invert
phrase pair list
target ||| source ||| alignment ||| ...
sort & score
sort, score & revert
sorted & scored phrase table half
source ||| target ||| alignment ||| ...
scored & reverted phr. table half
target ||| source ||| alignment ||| ...
merge
sort
full phrase table (text format)
source ||| target ||| fwd. & bwd. scores ||| ...
merge
scored & sorted phrase table half
source ||| target ||| bwd. scores ||| ...
prune
pruned phrase table
source ||| target ||| fwd. & bwd. scores ||| internal word alignment ||| ...
binarise
binary phrase table

Figure 1. Conventional Phrase Table Construction

tate the accumulation of marginals.This approach requires sorting the list of extracted phrase pairs at least twice: once to obtain joint and marginal counts for estimation of the forward translation probabilities, and once to calculate the marginals for the backward probabilities. In practice, forward and backward scoring take place in parallel, as shown in Figure 1.

The resulting phrase tables often have considerable levels of noise, due to misaligned sentence pairs or alignment errors at the word level. Phrase table *pruning* removes entries of dubious quality. Even with pruning, conventional phrase tables built from large amounts of parallel data are often too large to be loaded and stored completely in memory. Therefore, various binary phrase table implementations were developed in *Moses* over the years, providing access to disk-based data base structures (Zens and Ney, 2007)[1] or using compressed representations that can be mapped into memory and "unpacked" on demand (Junczys-Dowmunt, 2012).

[1]The original implementation by R. Zens (*PhraseDictionaryBinary*) has recently been replaced in *Moses* by *PhraseDictionaryOnDisk* (H. Huang, personal communication).

Due to the way they are built, conventional phrase tables for *Moses* are fundamentally static in nature: they cannot be updated easily without repeating the entire costly creation process.

2. Phrase tables with on-demand sampling

As an alternative to pre-computed phrase tables, Callison-Burch et al. (2005) suggested the use of suffix arrays (Manber and Myers, 1990) to index the parallel training data for full-text search, and to create phrase table entries on demand at translation time, by sampling in the bitext occurrences of each source phrase in question, extracting counts and statistics as necessary.

A suffix array over a corpus $\langle w_1, \ldots, w_n \rangle$ is an array $\langle 1 \ldots n \rangle$ of all token positions in that corpus, sorted in lexicographic order of the token sequences that start at the respective positions. Figure 2 shows a letter-based suffix array over the word 'suffixarray'. For bitext indexing for MT, we index at the word level.

7	suffix	array
10	suffixarr	ay
3	su	ffixarray
4	suf	fixarray
5	suff	ixarray
9	suffixar	ray
8	suffixa	rray
1		suffixarray
2	s	uffixarray
6	suffi	xarray
11	suffixarra	y

Figure 2. Letter-based suffix array over the word 'suffixarray'

Given a suffix array and the underlying corpus, we can easily find all occurrences of a given search sequence by performing a binary search in the array to determine the first item that is greater or equal to the search sequence, and a second search to find the first item that is strictly greater. Every item in this subrange of the array is the start position of an occurrence of the search sequence in the corpus. From this pool of occurrences, we extract phrase translations for a reasonably large sample using the usual phrase extraction heuristics.

Lopez (2007, 2008) explored this approach in detail in the context of *hierarchical phrase-based translation* (Chiang, 2007). Schwartz and Callison-Burch (2010) implemented Lopez's methods in the *Joshua* decoder (Li et al., 2009). Suffix array-based translation rule extraction is also used in *cdec* (Dyer et al., 2010). However, until recently (Germann, 2014), no efficient, working implementation of sampling phrase tables was available in the *Moses* decoder. The purpose of this article is to document this implementation in detail, and to present results of empirical evaluations that demonstrate that sampling phrase tables are an attractive, efficient, and competitive alternative to conventional phrase tables for phrase-based SMT.

The apparent lack of interest in sampling phrase tables in the phrase-based SMT community may have been partly due to the fact that naïve implementations of the approach tend perform worse than conventional phrase tables. To illustrate this point, we repeat in Table 1 the results of a comparison of systems from Germann (2014). Several German-to-English systems were constructed with conventional and sampling phrase tables. All systems were trained on ca. 2 million parallel sentence pairs from Europarl (Version 7) and the News Commentary corpus (Version 9), both available

#	method	low	high	median	mean	95% conf. interval[a]	runs
1	**precomp., Kneser-Ney smoothing**	18.36	18.50	18.45	18.43	17.93 – 18.95	10
2	**precomp., Good-Turing smoothing**	18.29	18.63	18.54	18.52	18.05 – 19.05	10
3	**precomp., Good-Turing smoothing, filtered**[b]	18.43	18.61	18.53	18.53	18.04 – 19.08	10
4	**precomp., no smoothing**	17.86	18.12	18.07	18.05	17.58 – 18.61	10
5	**max. 1000 smpl., no sm., no bwd. prob.**	16.70	16.92	16.84	16.79	16.35 – 17.32	10
6	**max. 1000 smpl., no sm., with bwd. prob.**	17.61	17.72	17.69	17.68	17.14 – 18.22	8
7	**max. 1000 smpl., α = .05, with bwd. prob.**[c]	18.35	18.43	18.38	18.38	17.86 – 18.90	10
8	**max. 1000 smpl., α = .01, with bwd. prob.**	18.43	18.65	18.53	18.52	18.03 – 19.12	10
9	**max. 100 smpl., α = .01, with bwd. prob.**	18.40	18.55	18.46	18.46	17.94 – 19.00	10

table adapted from Germann (2014)

[a] computed via bootstrap resampling for the median system in the group.
[b] top 100 entries per source phrase selected according to $p\left(\mathbf{t} \mid \mathbf{s}\right)$.
[c] α: one-sided confidence level of the Clopper-Pearson confidence interval for the observed counts.

Table 1. Bleu *scores (de → en) with different phrase score computation methods.*

from the web site of the 2014 *Workshop on Statistical Machine Translation* (WMT).[2] They were tuned on the `NewsTest 2013` data set, and evaluated on the `NewsTest 2014` data set from the Shared Translation Tasks at WMT-2013 and WMT-2014, respectively. The systems differ in the number of feature functions used (with and without backwards phrase-level translation probabilities) and smoothing methods applied. No lexicalized reordering model was used in these experiments.

Each system was tuned 8-10 times in independent tuning runs with Minimum Error Rate Training (MERT; Och, 2003). Table 1 shows low, high, median, and mean scores over the multiple tuning runs for each system.

The first risk in the use of sampling phrase tables is that it is tempting to forfeit the backwards phrase-level translation probabilities in the scoring. The basic sampling and phrase extraction procedure produces source-side marginal and joint counts over a sample of the data, but not the target-side marginal counts necessary to compute $p\left(\textit{source} \mid \textit{target}\right)$. These backwards probabilities are, however, important indicators of phrase-level translation quality, and leaving them out hurts performance, as illustrated by a comparison of Lines 2 and 5 in Tab. 1 (standard setup vs. naïve implementation of the sampling approach without backward probabilities and smoothing).

While it is technically possible to "back-sample" phrase translation candidates by performing the sampling and gathering of counts inversely for each phrase translation candidate, this would greatly increase the computational effort required at translation time, and slow down the decoder. A convenient and effective short-cut, however, is to simply scale the target-side global phrase occurence counts of each translation

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

candidate by the proportion of sample size to total source phrase count:

$$\mathrm{p}\,(source\,|\,target) \approx \frac{\text{joint phr. count in sample}}{\text{total target phr. count}} \cdot \frac{\text{total source phr. count}}{\text{\# of source phr. sampled}} \tag{1}$$

As Line 6 in Tab. 1 shows, this method narrows the performance gap between conventional systems and sampling phrase tables, althout it does not perform as well as "proper" computation of the backwards probabilities (cf. Line 4 in the table).

The second disadvantage of the sampling approach is that it cannot use the standard smoothing techniques used to compute smoothed phrase table scores in conventional phrase tables, i.e. Good-Turing or Kneser-Ney, as these require global information about the phrase table that is not available when sampling. The results in Lines 4 and 6 (vs. Line 2) confirm the finding by Foster et al. (2006) that phrase table smoothing improves translation quality.

One particular problem with maximum likelihood (ML) estimates in the context of translation modeling is the over-estimatation of the observations in small samples. The smaller the sample, the bigger the estimation error. Since the decoder is free to choose the segmentation of the input into source phrases, it has an incentive to pick long, rare phrases. The smaller sample sizes result in bigger over-estimation of the true translation probabilities. This in turn leads to higher model scores, which is what the decoder aims for. Alas, in this case higher model scores usually do not mean higher translation quality — ML estimates introduce modelling errors. Smoothing dampens this effect.

In lieu of the established smoothing techniques, we counteract the lure of small sample sizes by replacing maximum likelihood estimates with the lower bound of the binomial confidence interval (Clopper and Pearson, 1934) for the observed counts in the actual sample, at an appropriate level of confidence.[3] Figure 3 shows the "response curve" of this method for a constant success rate of 1/3, as the underlying sample size increases. In practice, a confidence level of 99% appears to be a reasonable choice: in our German-to-English experiments, using the lower bound of the binomial confidence interval at this level brought the BLEU performance of the system with a sampling phrase table back to the level of decoding with a conventional phrase table (cf. Line 8 vs. Line 2 in Tab. 1).

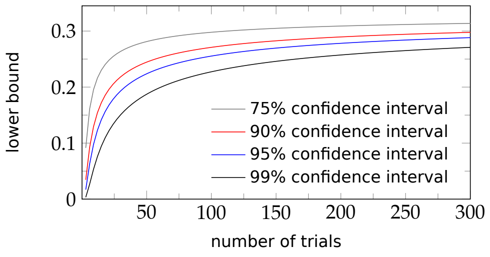

Figure 3. Lower bound of binomial confidence interval for success rate $\frac{1}{3}$

[3] An alternative is the use of additional features that keep track of quantized raw joint phrase counts, e.g., how many phrase translations used in a translation hypothesis were actually observed at most once, twice, three times, or more frequently (Mauser et al., 2007).

Another concern about sampling phrase tables is speed. After all, phrase extraction and assembly of phrase table entries on the fly do require more computation at translation time. However, caching of entries once created as well as multi-threaded sampling make the current implementation of sampling phrase tables in *Moses* very competitive with their alternatives.

A comparision of translation times for a large French–English system with sampling phrase tables vs. the compressed phrase tables of Junczys-Dowmunt (2012) (*CompactPT*) is given in Tab. 3 (Sec. 7) below. *CompactPT* is the fastest phrase table implementation available in *Moses* for translation in practice.

3. Lexicalised reordering model

Lexicalized reordering models improve the quality of translation (Galley and Manning, 2008). Sampled lexicalised reordering models were not available for the work presented in Germann (2014), but have been implemented since. Our sampling procedure keeps track of the necessary information for hierarchical lexicalized reordering (Galley and Manning, 2008) and communicates this information to the lexicalised reordering model.

4. Dynamic updates

One special feature of the sampling phrase table implementation in *Moses* is that it allows to add parallel text dynamically through an RPC call when moses is run in server mode. This is useful, for example, when *Moses* serves as the MT back-end in an interactive post-editing scenario, where bilingual humans post-edit the MT output. Dynamic updates allow immediate exploitation of the newly created high-quality data to improve MT performance on the spot.

To accommodate these updates, the phrase table maintains two separate bitexts: the memory-mapped, static *background* bitext whose build process was just described, and a dynamic *foreground* bitext that is kept entirely in-memory. The phrase table's built-in feature functions can be configured to compute separate scores for foreground and background corpus, or to simply pool the counts. Details for use of this feature are available in the *Moses* online documentation.[4]

5. Building and using sampling phrase tables in *Moses*

5.1. *Moses* compilation

Compile *Moses* as usual, but with the switch `--with-mm`:[5]

```
./bjam --with-mm --prefix=...
```

[4]`http://www.statmt.org/moses`

[5]Suffix-array based sampling phrase tables are scheduled to be included with the standard built soon.

The binaries `mtt-build`, `symal2mam`, and `mmlex-build` will be placed in the same directory as the `moses` executable.

5.2. Binarizing the word-aligned parallel corpus

Binarisation converts the corpus from a text representation into large arrays of 32-bit word IDs, and creates the suffix arrays. Word alignment information is also converted from a text representation (`symal` output format) to a binary format. In addition, a probabilistic word translation lexicon is extracted from the word-aligned corpus and also stored in binary format. All files are designed to be mapped directly in to memory for fast loading.

Let *corpus* be the base name of the parallel corpus. The tags *src* and *trg* are language tags that identify the source and the target language. Normally, these tags are mnemonic tags such as *en, fr, de,* etc.

- *corpus.src* is the source side of the corpus (one sentence per line, tokenized);
- *corpus.trg* is the respective target side in the same format;
- *corpus.src-trg.symal* is the word alignment between the two in the format produced by the `symal` word alignment symmetriser;
- */some/path/* is the path where the binarized model files will be stored. It must exist prior to running the binarizers. The path specification may include a file prefix *bname.* for the individual file names, in which case */some/path/bname.* should be used instead of */some/path/* in all steps.

Binarisation consists of four steps, the first three of which can be run in parallel. The

Step 1: binarise source side: `mtt-build < corpus.src -i -o /some/path/src`
Step 2: binarise target side: `mtt-build < corpus.trg -i -o /some/path/trg`
Step 3: binarise word alignments

```
symal2mam < corpus.src-trg.symal /some/path/src-trg.mam
```

Step 4: produce a word lexicon for lexical scoring

```
mmlex-build corpus. src trg -o /some/path/src-trg.lex
```

Steps 1 and 2 will produce 3 files each: a map from word strings to word IDs and vice versa (**.tdx*), a file with the binarized corpus (**.mct*), and the corresponding suffix array (**.sfa*). Steps 3 and 4 produce one file each (*.mam and *.lex, respectively).

5.3. Setting up entries in the `moses.ini` file

In the section `[feature]`, add the following two entries.

```
LexicalReordering name=DM0 type=hier-mslr-bidirectional-fe-allff
Mmsapt name=PT0 lrfunc=DM0 path=/some/path/ L1=src L2=trg sample=1000
```

Note that the value of the `path` parameter must end in '/' or '.', depending on whether it points to a directory or includes a file name prefix. The value of the parameter `lrfunc` must match the name of the lexical reordering feature.

5.4. Setting up sampling phrase tables in EMS

In *Moses*'s *Experiment Management System* (EMS), the use of sampling phrase tables can be specified by adding the following two lines to the EMS configuration file.

```
mmsapt = "sample=1000"
binarize-all = $moses-script-dir/training/binarize-model.perl
```

6. Configuring the phrase table

The phrase table implementation offers numerous configuration options. Due to space constraints, we list only the most important ones here; the full documentation can be found in the online *Moses* documentation at `http://statmt.org/moses`. All options can be specified in the phrase table's configuration line in `moses.ini` in the format `key=value`. Below, the letter '*n*' designates numbers in $\mathbb{N}$, '*f*' floating point numbers, and '*s*' strings.

6.1. General Options

sample=*n* the maximum number of samples considered per source phrase.

smooth=*f* the "smoothing" parameter. A value of 0.01 corresponds to a 99% confidence interval.

workers=*n* the degree of parallelism for sampling. By default (`workers=0`), all available cores are used. The phrase table implements its own thread pool; the general *Moses* option `threads` has no effect here.

cache=*n* size of the cache. Once the limit is reached, the least recently used entries are dropped first.

ttable-limit=*n* maximum number of distinct translations to return.

extra=*s* path to additional word-aligned parallel data to seed the foreground corpus for use in an interactive *dynamic* scenario where phrase tables can be updated while the server is running. This use case is explained in more detail in Germann (2014).

6.2. Feature Functions

Currently, word-level lexical translation scores are always computed and provided. Below we list some core feature scores that the phrase table can provide. A comprehensive list including experimental features is provided online at `http://statmt.org/moses`.

pfwd=g[+] forward phrase-level translation probability. If `g+` is specified, scores are computed and reported separately for the static background and the dynamic foreground corpus. Otherwise, the underlying counts are pooled.

pbwd=g[+] backwards phrase-level translation probability with the same interpretation of the value specified as for `pfwd`.

source	**sentence pairs**	**French tokens**	**English tokens**
CommonCrawl	3.2 M	86 M	78 M
EuroParl	2.0 M	58 M	52 M
Fr–En Gigaword	21.4 M	678 M	562 M
News Commentary	0.2 M	6 M	5 M
UN	12.3 M	367 M	318 M
Total for TM training	**39.1 M**	**1,185 M**	**1,016 M**
News data for LM training	**140.0 M**		**2,874 M**

Table 2. Corpus statistics for the parallel WMT-2015 French-English training data.

lenrat={0|1} phrase length ratio score (off/on). Phrase pair creation is modelled as a Bernoulli process with a biased coin: 'heads': produce a word in *L1*, 'tails': produce a word in *L2*. The bias of the coin is determined by the ratio of the lengths (in words) of the two sides of the training corpus. This score is the log of the probability that the phrase length ratio is no more extreme (removed from the mean) than observed.

rare=*f* rarity penalty: $\frac{f}{f+j}$, where j is the phrase pair joint count. This feature is always computed on the pooled counts of foreground and background corpus.

prov=*f* provenance reward: $\frac{j}{f+j}$. This feature is always computed separately for foreground and background corpus.

7. Performance on a large dataset

Table 3 shows build times and translation performance of two systems built with the large French–English data set available for the WMT-2015 shared translation task (cf. Tab. 2). The first system uses a pruned conventional phrase table binarized as a compact phrase table (Junczys-Dowmunt, 2012) (tuning was performed with an unpruned, filtered in-memory phrase table); the other system uses a sampling phrase table. The systems were tuned on 760 sentence pairs from the `newsdiscussdev2015` development set and evaluated on the `newsdiscusstest2015` test set.

For technical reasons, we were not able to run the **build processes** on dedicated machines with identical specifications; build times reported are therefore only approximate numbers. To give the conventional phrase table construction process the benefit of the doubt, the data binarization for the sampling phrase table was performed on a less powerful machine (8 cores) than conventional phrase table construction (24-32 cores), although not all steps in the process can utilize multiple cores. Nevertheless, even under these slightly unfair conditions the time savings of the sampling approach are obvious. The **translation speed** experiments were performed with cube pruning (pop limit: 1000) on the same 24-core machine with 148GB of memory, translating the test set of 1500 sentences (30,000 words) in bulk using all available cores on

	conventional system	**sampling phrase tables**
phrase table build time	$\gg$ 20 hrs.	ca. 1h 30m
Model features:	total: 28	total: 18
• word penalty	yes	yes
• phrase penalty	yes	yes
• distortion distance	yes	yes
• language model	5-gram Markov model	5-gram Markov model
• TM: phrase transl.	forward, backward	forward, backward
	w/ Good-Turing sm.	lower bound of 99% conf. interv.
• TM: lexical transl.	forward, backward	forward, backward
• rare counts	6 bins: 1/2/3/4/6/10	rarity penalty
• lex. reord. model	hierarchical-fe-mslr-all-ff	hierarchical-fe-mslr-all-ff
• phrase length ratio	no	yes

Evaluation (`newsdiscusstest2015`) — 3 independend tuning runs

	run	BLEU	95% conf. interval via boostrap resampling	run	BLEU	95% conf. interval via boostrap resampling
batch MIRA	#1	33.16	32.07 – 34.21	#1	33.16	31.96 – 34.27
	#2	33.42	32.42 – 34.52	#2	32.89	31.74 – 34.04
	#3	33.30	32.16 – 34.39	#3	33.12	32.03 – 34.20
MERT	#1	32.19	31.15 – 33.13	#1	34.25	33.11 – 35.37
	#2	32.93	31.90 – 34.08	#2	34.11	32.91 – 35.37
	#3	31.53	30.39 – 32.68	#3	33.80	32.69 – 34.90

translation speed

	unpruned	top30	sample=1000	sample=100
threads	8	24	24	24
wrds./sec. (sec./wrd.)	13 (0.075)	547 (0.002)	300 (0.003)	501 (0.002)
snts./sec. (sec./snt.)	0.7 (1.498)	27 (0.037)	15 (0.067)	25 (0.040)
BLEU (best system)	33.42	33.55	34.25	33.82

Table 3. Features used and translation performance for the WMT15 fr–en experiments.

the machine. Prior to the start of *Moses*, all model files were copied to `/dev/null` to push them into the operating system's file cache. Due to race conditions between threads, we limited the number of threads to 8 for the legacy system with the unpruned phrase table.

Notice that in terms of BLEU scores, the two systems perform differently with different tuning methods. The lower performance of MERT for the conventional system with 28 features is not surprising: it is well known that MERT tends to perform poorly when the number of features exceeds 20. That MIRA fares worse than MERT for the sampling phrase tables may be due to a sub-optimal choice of MIRA's meta-parameters (cf. Hasler et al., 2011 for details on MIRA's meta-parameters).

8. Conclusion

We have presented an efficient implementation of sampling phrase tables in *Moses*. With the recent integration of hierarchical lexicalized reordering models into the approach, sampling phrase tables reach the same level of translation quality while approaching *CompactPT* in terms of speed. In addition, sampling phrase tables offer the following advantages that make them an attractive option both for experimentation and research, and for use in production environments:

- They are much faster to build.
- They offer flexibility in the choice of feature functions used. Feature functions can be added or disabled without creating the need to re-run the entire phrase table construction pipeline.
- They have a lower memory footprint. It is not necessary to filter or prune the phrase tables prior to translation.

9. Availability

Sampling phrase tables are included in the master branch of *Moses* in the *Moses* github repository at `http://github.com/moses-smt/mosesdecoder.git`.

Acknowledgements

This work was supported by the European Union's *Horizon 2020* research and innovation programme (H2020) under grant agreements 645487 (MMT) and 645452 (QT21). It extends work supported by the EU's *Framework 7* (FP7) program under grant agreements 287688 (MateCat), 287576 (CasMaCat), and 287576 (ACCEPT).

Bibliography

Callison-Burch, Chris, Colin Bannard, and Josh Schroeder. Scaling Phrase-Based Statistical Machine Translation to Larger Corpora and Longer Phrases. In *43rd Annual Meeting of the Association for Computational Linguistics (ACL '05)*, pages 255–262, Ann Arbor, Michigan, 2005.

Chiang, David. Hierarchical Phrase-Based Translation. *Computational Linguistics*, 33(2):1–28, 2007.

Clopper, C.J. and E.S. Pearson. The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika*, 1934.

Dyer, Chris, Adam Lopez, Juri Ganitkevitch, Jonathan Weese, Ferhan Ture, Phil Blunsom, Hendra Setiawan, Vladimir Eidelman, and Philip Resnik. cdec: A Decoder, Alignment, and Learning Framework for Finite-State and Context-Free Translation Models. In *Proceedings of the ACL 2010 System Demonstrations*, pages 7–12, Uppsala, Sweden, 2010.

Foster, George F., Roland Kuhn, and Howard Johnson. Phrasetable Smoothing for Statistical Machine Translation. In *EMNLP*, pages 53–61, 2006.

Galley, Michel and Christopher D. Manning. A Simple and Effective Hierarchical Phrase Reordering Model. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 848–856, Honolulu, Hawaii, 2008.

Germann, Ulrich. Dynamic Phrase Tables for Machine Translation in an Interactive Post-editing Scenario. In *Proceedings of the Workshop on Interactive and Adaptive Machine Translation*, pages 20–31, 2014.

Hasler, Eva, Barry Haddow, and Philipp Koehn. Margin Infused Relaxed Algorithm for Moses. *The Prague Bulletin of Mathematical Linguistics*, 96:69–78, 2011.

Junczys-Dowmunt, Marcin. Phrasal Rank-Encoding: Exploiting Phrase Redundancy and Translational Relations for Phrase Table Compression. *Prague Bull. Math. Linguistics*, 98: 63–74, 2012.

Li, Zhifei, Chris Callison-Burch, Chris Dyer, Sanjeev Khudanpur, Lane Schwartz, Wren Thornton, Jonathan Weese, and Omar Zaidan. Joshua: An Open Source Toolkit for Parsing-Based Machine Translation. In *Fourth Workshop on Statistical Machine Translation*, pages 135–139, Athens, Greece, 2009.

Lopez, Adam. Hierarchical Phrase-Based Translation with Suffix Arrays. In *EMNLP-CoNLL*, pages 976–985, 2007.

Lopez, Adam. *Machine Translation by Pattern Matching*. PhD thesis, University of Maryland, College Park, MD, USA, 2008.

Manber, Udi and Gene Myers. Suffix Arrays: A New Method for On-line String Searches. In *Proceedings of the First Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '90, pages 319–327, Philadelphia, PA, USA, 1990. ISBN 0-89871-251-3.

Mauser, Arne, David Vilar, Gregor Leusch, Yuqi Zhang, and Hermann Ney. The RWTH Machine Translation System for IWSLT 2007. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT)*, 2007.

Och, Franz Josef. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Sapporo, Japan, 2003.

Schwartz, Lane and Chris Callison-Burch. Hierarchical Phrase-Based Grammar Extraction in Joshua: Suffix Arrays and Prefix Trees. *The Prague Bulletin of Mathematical Linguistics*, 93: 157–166, 2010.

Zens, Richard and Hermann Ney. Efficient Phrase-Table Representation for Machine Translation with Applications to Online MT and Speech Translation. In *Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL '07)*, pages 492–499, Rochester, New York, 2007.

Address for correspondence:
Ulrich Germann
ugermann@inf.ed.ac.uk
University of Edinburgh • 10 Crichton Street • Edinburgh, EH8 9AB, United Kingdom