Handling Technical OOVs in SMT

Mark Fishel and **Rico Sennrich**
Institute of Computational Linguistics
University of Zurich
Binzmühlestr. 14
CH-8050 Zürich
{fishel,sennrich}@cl.uzh.ch

Abstract

We present a project on machine translation of software help desk tickets, a highly technical text domain. The main source of translation errors were out-of-vocabulary tokens (OOVs), most of which were either in-domain German compounds or technical token sequences that must be preserved verbatim in the output. We describe our efforts on compound splitting and treatment of non-translatable tokens, which lead to a significant translation quality gain.

1 Problem Setting

In this paper we focus on statistical machine translation of a highly technical text domain: software help desk tickets, or put simply – bug reports. The project described here was a collaboration between the University of Zurich and Finnova AG and aimed at developing an in-domain translation system for the company's bug reports from German into English. Here we present a general description of the key project results, the main problems we faced and our solutions to them.

Technical texts like bug reports present an increased challenge for automatic processing. In addition to having a highly specific lexicon, there is often a large amount of source code snippets, form and database field identifiers, URLs and other "technical" tokens that have to be preserved in the output without translation – for example:

Ger:	siehe auch ecl_kd042_de_crm_basis MP-MAR-11, kapitel 9.2.1.1
Eng:	see also ecl_kd042_de_crm_basis MP-MAR-11, chapter 9.2.1.1

While these technical tokens need no translation, our baseline system also suffers from a large number of out-of-vocabulary tokens (OOVs) that should be translated. The concatenative morphology of German compounds is a classical problem for machine translation, as it leads to an increased vocabulary and exacerbates data sparsity (Koehn and Knight, 2003). In our case the problem is inflated due to the domain-specific compound terms like *Tabellenattribute* (table attribute) or *Nachbuchungen* (subsequent postings): many of these are not seen in the smaller in-domain parallel corpus and they are too specific to be present in general-domain corpora.

Technical tokens like URLs and alphanumeric IDs do not require translation and should be transferred into the output verbatim. However, since they are also unknown to the translation system, they still present a number of problems. They are often broken by tokenization and not restored properly by subsequent de-tokenization. Also, splitting a technical token into several parts might result in the internal order of those parts broken. Even tokens that are correctly preserved in their original form can cause problems: if they are unknown to the language model, the model strongly favours permutations of the output in which OOVs are grouped together.

In the following section we give a description of our project and baseline system. We then turn to the problem of OOVs, and focus on handling the technical tokens that require no translation in Section 3, and on compound splitting strategies in Section 4. Experimental results constitute Section 5.

2 Translating Help Desk Tickets

The aim of our project was to develop an in-domain translation system for translating help desk

Token Type	Regular Expression	Examples
DB and form field IDs	`[A-Z0-9][A-Z0-9_/-]*[A-Z0-9]`	BEG_DAT_BUCH
numbers	`-?[0-9]+([.'][0-9]+)?`	-124.30, 1'000
UNIX paths and URLs	`([^ ():]*/){2,}[^ ():]*`	/home/user/readme.txt
code with dots, e.g. java	`[^ :.]{2,}(\.[^ :.]{2,})+`	java.lang.Exception

Table 1: Examples of technical tokens and regular expressions for their detection.

tickets from German to English for use in a post-editing work-flow.

The company had a set of manual translations from the target domain, which enabled us to use statistical machine translation (SMT). The in-domain parallel corpus composed of these translations consisted of 227 000 parallel sentences (2.8 / 3.2 million German/English tokens). Additional monolingual English data for the same domain was also available (141 000 sentences, 1.9 million tokens). As a baseline we used the Moses framework (Koehn et al., 2007) with settings identical to the baseline of WMT shared tasks (Bojar et al., 2013).

To increase the vocabulary of the system we added some publicly available general-domain and out-of-domain parallel corpora: Europarl (Koehn, 2005), OPUS OpenSubtitles (Tiedemann, 2012) and JRC-Acquis (Steinberger et al., 2006). Each of these is at least 10 times bigger than our in-domain corpus. To prefer in-domain translations in case of ambiguity, we combined all the available corpora via instance weighting using TMCombine from the Moses framework (Sennrich, 2012).

Despite the vast amount of general-domain data, the improvement over an in-domain system is relatively small: from 21.9 up to 22.3 BLEU points.[1] This best confirms that our target domain is highly specific. In fact, general-domain data actually hurts translation performance if its size is greater and no domain adaptation is performed: a simple concatenation of the same corpora without weighting causes a drop in translation quality to 21.3 BLEU points.

A post-editing set-up with our translation system resulted in an average efficiency gain of 30% over a pure translation work-flow, raising the number of ticket translations per hour from 4.5 to 5.9. In the next sections, we describe further attempts to improve translation quality by addressing different types of OOVs in the system.

[1]Measured on a test set of 1000 randomly held-out sentences, detokenized and re-cased.

3 Preserving Technical Tokens

The main problems with technical tokens that do not require translation are preserving their orthography and internal order, and placing them at the correct position in a sentence.

Most of these tokens are highly regular, which means that they can be detected with regular expressions and handled separately. We designed a set of regular expressions for that purpose and tagged them with the type of tokens that they detect. Table 1 presents some examples of the regular expressions and detected tokens. 8.8% of the tokens are identified as "technical", with the largest group being upper-case database and form field IDs (4.0% of the tokens) and numbers (1.6% of the tokens).

We use XML mark-up to mark all technical tokens (consequently referred to as *masking*), and pass masked tokens unchanged through all components of our translation pipeline, i.e. the tokenizer, lowercaser, and the Moses decoder. While masking ensures that the masked tokens themselves are preserved, their position in the output is determined by the decoder. We observed that the n-gram language model that we use for decoding is poor at modelling the position of unknown words, preferring translation hypotheses where unknown words are grouped together, often at the beginning or end of the sentence.

As a solution to this issue, we change the translation pipeline as follows:

- the input text is tokenized and the detected technical tokens are reduced to a single constant token `__TECH__`.
- the translation is done on reduced text; the phrase table, lexical reordering and the language model are trained on corpora with reduced technical expressions.
- after the translation step, the reduced expressions are restored based on the input text and the word alignment between the input and the output, which is reported by the decoder.

This way, the original form of the technical tokens is preserved explicitly, and the feature functions of the translation pipeline do not have to deal with additional unknown input (the approach will be referred to as *1-token reduction*).

An alternative variant we explored is to represent each token sequence with its type (like `JAVA`, `DATE`, `URL`, etc.) instead of a single token `TECH`. A higher level of detail could be useful to model differences in word order between different kinds of technical tokens. Also, in case a sentence contains maskable tokens of different types, this reduces the number of duplicate tokens between which the model cannot discriminate (this alternative will be referred to as *type reduction*).

4 Compound splitting

The German language has a productive compounding system, which increases vocabulary size and exacerbates the data sparsity effect. Many compounds are domain-specific and are unlikely to be learned from larger general-domain corpora. Compound splitting, however, has the potential to also work on our in-domain texts.

We evaluate two methods of compound splitting. Koehn and Knight (2003) describe a purely data-driven approach, in which frequency statistics are collected from the unsplit corpus, and words are split so that the geometric mean of the word frequencies of its parts is maximized. Fritzinger and Fraser (2010) describe a hybrid approach, which uses the same corpus-driven selection method to choose the best split of a word among multiple candidates, but instead of considering all character sequences to be potential parts, they only consider those splits that are validated by a finite-state morphology tool.

The motivation for using the finite-state morphology is to prevent linguistically implausible splittings such as *Testsets* $\rightarrow$ *Test ETS*. We use the Zmorge morphology (Sennrich and Kunz, 2014), which combines the SMOR grammar (Schmid et al., 2004) with a lexicon extracted from Wiktionary.[2] With this hybrid approach, we only consider nouns for compound splitting; with the data-driven approach on the other hand we have no control over which word classes are split.

[2] `http://www.wiktionary.org`

Source:	erweiterung tabellen TX_VL und TXTSVL .
Reference:	extension of tables TX_VL and TXTSVL .
Masking:	extension of tables TX_VL TXTSVL and .
Reduction:	extension of tables TX_VL and TXTSVL .

Table 2: An example of the effect of reducing: the correct order of technical tokens is preserved.

5 Experiments and Results

We evaluated our experiments on a held-out in-domain test set. Translation quality is judged using the MultEval package (Clark et al., 2011) and its default automatic metrics (BLEU, TER and METEOR); the package implements the metrics and performs statistical significance testing to account for optimizer instability. We perform three independent tuning runs, and use 95% as the significance threshold. Statistically **non**-significant results are shown in italics. Since tokenization differs between experiments, we compare de-tokenized and re-cased hypothesis and reference translations.

As baseline, we use the weighted combination of in-domain and other corpora, described in Section 2. All modifications to tokenization and compound splitting are done on all included training corpora, both in-domain and others.

Masking the detected technical tokens yields large quality gains over default tokenization:

	BLEU	METEOR	TER
Baseline	22.3	26.1	62.2
Masking	25.1	27.6	56.8

The system with masking better matches the length of the reference translation than the baseline (99.5% vs. 103.7%); this can be attributed to the technical tokens being broken in the baseline and not fixed by the default de-tokenization.

The reduced representation of technical tokens brings a small improvement:

	BLEU	METEOR	TER
Just masking	25.1	27.6	56.8
1-token reduction	25.5	27.7	56.4
Type reduction	25.4	27.7	*56.6*

A manual inspection supports the hypothesis that the reduced representation improves word order for sentences with multiple OOVs; see Table 2 for an example. Representing the expressions with their type, however, does not seem to have any additional effect: statistically it is indistinguishable from 1-token reduction.

Compound splitting yields gains of 0.8–1 BLEU when evaluated separately from technical token reduction:

	BLEU	METEOR	TER
Just masking	25.1	27.6	56.8
Data-driven split	26.1	28.9	55.1
Hybrid split	25.9	28.6	55.4

In contrast to the results reported by Fritzinger and Fraser (2010), we observe no gains of the hybrid method over the purely data-driven method by Koehn and Knight (2003). We attribute this to the fact that domain-specific anglicisms such as *Eventhandling* (event handling) and *Debugmeldung* (debug message) are unknown to the morphological analyzer, but are correctly split by the data-driven method.

Finally, we obtain the best system by combining masking, 1-token reduction and data-driven segmentation.

	BLEU	METEOR	TER
Just masking	25.1	27.6	56.8
1-token reduction	25.5	27.7	56.4
Data-driven split	26.1	28.9	55.1
Full combination	26.5	29.0	54.1

To conclude, we have shown that the modelling of OOVs has a large impact on translation quality in technical domains with high OOV rates. Overall we observed an improvement of 4.2 BLEU, 2.9 METEOR and 8.1 TER points over the baseline.

In this paper, we focused on two types of OOV tokens: German compounds that can be split into their components, and technical tokens that need no translation. While our modelling of both these types was successful both individually and in combination, in the general case the handling of different types of OOVs are not necessarily independent steps. Also, additional strategies for handling OOVs may be required in other domains and language pairs, e.g. transliteration of named entities. Robustly choosing the right strategy for each OOV token independently of the domain could be the target of future research.

References

Bojar, Ondřej, Christian Buck, Chris Callison-Burch, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2013. Findings of the 2013 Workshop on Statistical Machine Translation. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 1–44, Sofia, Bulgaria.

Clark, Jonathan H, Chris Dyer, Alon Lavie, and Noah A Smith. 2011. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *Proceedings of the 49th ACL*, pages 176–181, Portland, Oregon, USA.

Fritzinger, Fabienne and Alexander Fraser. 2010. How to avoid burning ducks: Combining linguistic analysis and corpus statistics for German compound processing. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 224–234, Uppsala, Sweden.

Koehn, Philipp and Kevin Knight. 2003. Empirical methods for compound splitting. In *Proceedings of the 10th EACL*, pages 187–193, Budapest, Hungary.

Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: open source toolkit for statistical machine translation. In *Proceedings of the 45th ACL*, pages 177–180, Prague, Czech Republic.

Koehn, Philipp. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of MT Summit X*, volume 5, pages 79–86.

Schmid, Helmut, Arne Fitschen, and Ulrich Heid. 2004. A German Computational Morphology Covering Derivation, Composition, and Inflection. In *Proceedings of the 4th LREC*, pages 1263–1266, Lisbon, Portugal.

Sennrich, Rico and Beat Kunz. 2014. Zmorge: A German morphological lexicon extracted from Wiktionary. In *Proceedings of the 9th LREC*, (in print), Reykjavik, Iceland.

Sennrich, Rico. 2012. Perplexity minimization for translation model domain adaptation in statistical machine translation. In *Proceedings of the 13th EACL*, pages 539–549, Avignon, France.

Steinberger, Ralf, Bruno Pouliquen, Anna Widiger, Camelia Ignat, Tomaz Erjavec, Dan Tufis, and Dániel Varga. 2006. The JRC-Acquis: A multilingual aligned parallel corpus with 20+ languages. In *Proceedings of the 5th LREC*, pages 2142–2147, Genoa, Italy.

Tiedemann, Jörg. 2012. Parallel data, tools and interfaces in OPUS. In *Proceedings of the 8th LREC*, pages 2214–2218, Istanbul, Turkey.