Dictionary-based Domain Adaptation of MT Systems without Retraining

Rudolf Rosa, Roman Sudarikov, Michal Novák, Martin Popel, Ondřej Bojar
Charles University in Prague, Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics
Malostranské náměstí 25, Prague, Czech Republic
{rosa,sudarikov,mnovak,popel,bojar}@ufal.mff.cuni.cz

Abstract

We describe our submission to the IT-domain translation task of WMT 2016. We perform domain adaptation with dictionary data on already trained MT systems with no further retraining. We apply our approach to two conceptually different systems developed within the QTLeap project: TectoMT and Moses, as well as Chimera, their combination. In all settings, our method improves the translation quality. Moreover, the basic variant of our approach is applicable to any MT system, including a black-box one.

1 Introduction

In this paper, we describe our work on domain adaptation of machine translation systems, performed in close collaboration with numerous partners within the QTLeap project.[1] The project focuses on high-quality translation for the IT domain, and our systems were submitted to the IT-domain translation task of the First Conference on Machine Translation (WMT16).[2] The experiments relate to our previous work on domain adaptation (Rosa et al., 2015), in which we also surveyed and evaluated common domain adaptation techniques.

The aim of our work is to find a way to perform domain adaptation of an already trained MT system without having to retrain it, which may be a useful ability for reasons discussed in § 2. We focus on forced translation of domain-specific entities according to a bilingual lexicon, as described in § 3. We explore several methods based on preprocessing and postprocessing of the data before and after processing them by the MT system, and provide both system-specific and system-independent approaches.

We employ the MT systems used and further developed by us and our partners within the QTLeap project, namely Moses (Koehn et al., 2007), TectoMT (Žabokrtský et al., 2008), and their combination Chimera (Bojar et al., 2013). We briefly describe the systems in § 4. In § 5, we evaluate our domain-adaptation methods (as well as the standard method of retraining the system with all available data) applied to these MT systems for translation from English to Czech (EN→CS), Spanish (EN→ES), Dutch (EN→NL), and Portuguese (EN→PT).

2 Motivation

A quite typical situation in domain adaptation of an MT system is as follows: there is an MT system trained on large general-domain data, and there is a small amount of parallel data from the target domain, often in the form of a dictionary rather than parallel sentences.

In such a case, the standard solution is to add the in-domain data to the general-domain data and retrain the MT system; or, if there is support from the system, to train a secondary in-domain translation model (phrase table) and add it to the system (which may require retuning the system). However, in this work, we investigate the options of performing domain adaptation of the system without having to train (or tune) anything.

There is a range of reasons why one would avoid retraining the system for the specific domain, some of which we list below.

Training costs The simplest reason is that the costs of retraining the system might be impractical or even prohibitive, be it costs in terms of computational power, money, or time. This issue may be-

[1] http://qtleap.eu

[2] http://www.statmt.org/wmt16/it-translation-task.html

Proceedings of the First Conference on Machine Translation, Volume 2: Shared Task Papers, pages 449–455,
Berlin, Germany, August 11-12, 2016.

come even more pronounced with the modern neural MT systems, which can take weeks to train.[3]

Running costs In case there is a number of separate domains for which the MT system must be adapted, even the costs of operating a number of separate MT systems may become significant. This situation may occur e.g. in a translation company which uses a lot of domain-specific or even client-specific glossaries. Operating only one system with lightweight domain adaptation done in preprocessing and/or postprocessing might save a lot of costs.

Unsuitable data In some cases, the in-domain data itself may not be suitable for standard MT processing. Dictionary-like and/or small data cannot be used to reliably estimate translation probabilities, which may lead to the in-domain data having only a low influence on the MT system. These problems and possible solutions are also discussed by Daumé III (2009).

Black-box scenario This is a rather theoretical situation in the research area, but in practice it constitutes a real and common mode of operation. Both individual and business users often simply use a trained MT system as a "black box", without the ability or possibility to retrain it or directly modify it. Such users then have no other option than to resort to domain adaptation methods that rely only on preprocessing and postprocessing.

Dependence on non-retrainable tools Many MT systems consist of several components, some of which may require specific data for training, not available for the target domain, or they may even be rule-based and thus only adaptable through a certain amount of manual labor. A prominent example is the TectoMT system, which relies, among other, on morphological taggers and syntactic parsers to analyze the input sentences, which are typically trained on general-domain data (mostly news) and are rather hard to adapt to domains that differ significantly lexically or even structurally. However, even simple tools, such as the standard Moses tokenizer, may need to be adjusted to the target domain. In the IT domain specifically, we frequently observe structures such as URLs, file paths, computer commands, or chains of menu items, which are rather rare or even non-existent in the general-domain texts, and thus can greatly confuse the analyzers; this is especially true for TectoMT, where such unexpected structures cause significant problems already to the morphological tagger and dependency parser, leading to quite unpredictable results in the subsequent linguistic processing. In such cases, a careful preprocessing may be able to adjust the input texts to better resemble general-domain texts, hiding "surprising" structures from the system.

[3]That said, if the training is done with online learning (or mini-batches) a simple domain-adaptation technique is to continue training on in-domain data, which is much faster.

3 Method

The general principle of our method is to force-translate some domain-specific expressions according to a gazetteer (a domain-specific bilingual lexicon), without modifying our MT system.

Based on an error analysis on the Batch2 part of the provided in-domain data, we target named entities from the IT domain that need to be translated or localized, such as menu items, button names, their sequences, and messages. These are expected to appear in a fixed form, which allows us to apply a technique of directly matching the expressions from a gazetteer in the surface source text and replacing them by their equivalents in the target language.[4] As our gazetteer, we use the dictionary available from the website of the task, constructed automatically mainly from localization files of various software products.

The crucial task is to identify the source expressions to be force-translated (§ 3.1). The technical implementation of the forced translation can then be either independent of the MT system (§ 3.2), or, if the system has support for forced translations, a system-specific method can be used (§ 3.3, § 3.4).

We implemented the methods in the Treex NLP framework of Popel and Žabokrtský (2010).[5]

3.1 Identification of domain-specific entities

This is the most complex stage of the whole process. Lexicon items are matched in the source tokenized text and the matched items, which can possibly span several neighboring tokens, are marked for forced translation.

In the initialization stage, the source-language part of the lexicon is loaded and structured in a word-based trie to reduce time complexity of the text search. In the current implementation, if an

[4]This means that only the base forms contained in the gazetteer are processed; expressions in a different form are not handled.

[5]`http://ufal.mff.cuni.cz/treex`

expression appears more than once in the source gazetteer list, only its first occurrence is stored, regardless what its translation is.[6]

The trie is then used to match the expressions from the gazetteer in the source text. As they might overlap, each matched expressions is assigned a score estimating the extent to which it is a named entity, using a heuristic scorer:

- +10 if it starts with a capital letter; -10 if not
- +10 if the corresponding gazetteer item starts with a capital letter; -50 if not
- +2 if it matches the gazetteer case-sensitively
- +1 if all its words start with a capital letter; -1 if not
- -50 if it spans the first word of the sentence; +1 if not
- -50 if its last word is "menu"
- -100 if it contains only non-alphabetical characters

The resulting score is then multiplied by the number of tokens in the matched expression.

The matches with positive score are ordered by the score and filtered to get non-overlapping matches, taking those with higher score first. Neighboring entities are then collapsed into one.[7]

The translation of each entity is then constructed as a concatenation of the translations of the source entities according to the lexicon, and the entity is marked in one of the ways described in the following sections.

3.2 XXX placeholders

A simple approach is to replace the matched entities by unique placeholders (such as "xxxitemaxxx", "xxxitembxxx"...), storing the corresponding translations into a text file together with their assigned placeholders.

The preprocessed text is then passed through the MT system; if sufficiently complex placeholders are constructed, it is safe to assume that they will constitute out-of-vocabulary items for the MT system and will pass through unchanged (that is, unless the system has a policy of dropping OOVs).

Finally, the translated text is postprocessed, replacing each placeholder with the corresponding translation from the text file.

This approach is independent of the MT system, and can even be used in a black-box scenario. However, introducing a large number of OOVs may negatively influence the performance of the MT system, as it forces it to use a very limited linear context around the placeholders.

3.3 Moses XML annotation

Moses supports XML markup for marking forced translations of some parts of the sentence.[8] For instance, in the sentence "Click the icon, then select Shut Down.", we can suggest to translate "Shut Down" into Czech as "Vypnout":

```
Click the icon, then select
<item translation="Vypnout"
  prob="0.8">Shut Down</item>.
```

The XML annotation feature is enabled in the decoder by using the `-xml-input` switch, instructing the decoder to do one of the following (based on the value of the switch):

- to treat the XML markup as part of the sentence (`pass-through`) – the default,
- to strip the XML markup (`ignore`),
- to make the suggested translation compete with phrase table choices (`inclusive` or `constraint`),
- or to use only the suggested translation, ignoring all phrases that overlap with the annotated span (`exclusive`).

We always used only one forced translation for each entity, and the `exclusive` setting. However, Moses supports listing a set of suggested translations together with translation probabilities, leaving it up to the decoder to choose the best translation in the given context. We leave this, together with experimenting with the `inclusive` setting, for future research.

3.4 TectoTM/Treex wild attributes

Unlike Moses, which operates on plaintext only, TectoMT uses a structured layered representation of the texts, which makes it easy to add a new annotation layer specifying the translations. The easiest is to use a set of general-purpose Treex attributes, called *wild attributes*.

The preprocessing consists of replacing each matched entity with a placeholder (we used the word "Menu" as a placeholder) and storing the translation into a wild attribute of the entity. This

[6] Therefore, the performance of gazetteer matching machinery depends on the ordering of the entries in the gazetteer.

[7] The entities are collapsed also when they are separated by a > symbol; the separators are retained in the forced translations. This measure is aimed at translation of menu items and button labels sequences, which frequently appear in the IT domain data.

[8] `http://www.statmt.org/moses/?n=Advanced.Hybrid`

is done just after tokenization of the source text, so that for subsequent analysis steps, such as linguistic taggers and parsers, the entity already appears as one token. Moreover, we assume that using a common word for the placeholder makes the sentence even more fluent and easy to process for the analyzers.

The specified translation is forced in the transfer step, in which the decoder checks for the wild attribute, and, if present, uses its contents to generate the translation of the token instead of its translation models.

3.5 Forced non-translations

For TectoMT, we use an additional preprocessing step, which we call *forced non-translations*: we enforce certain special structures, frequent in the IT domain, to remain untranslated (namely URLs, e-mail addresses, Windows and Unix paths and file names, and shell commands). In principle, this is the same thing as forced translations, but based on regular expressions rather than a dictionary, since we identify these entities based on structural rather than lexical cues.

Although preliminary experiments with TectoMT and Batch2 dataset indicated a significant potential of this step (up to +0.4 BLEU), further evaluation revealed that this is mostly specific to this particular setup, as Batch2 contains a large number of these structures, and TectoMT greatly benefits from the single-token placeholder analysis of these structures. The Moses tokenizer can be setup to tokenize URLs and e-mails as single tokens, and even in case of multi-token analysis the Moses decoder is not confused so much as the TectoMT analysis steps.

Therefore, we omit an analysis of performance of the forced non-translations from the evaluation section: we simply always apply it in the TectoMT system, but never in the Moses system.[9]

4 MT Systems

We use two systems, Moses (Koehn et al., 2007) and TectoMT (Žabokrtský et al., 2008), as well as their combination Chimera (Bojar et al., 2013); see also a more detailed description of the Moses and TectoMT systems within the QTLeap project by Gaudio et al. (2016) in these proceedings.

[9]This holds even for the Chimera combination, i.e. this method is applied in its TectoMT component but not in the Moses component.

All of our systems are "constrained", i.e. trained and tuned using only the general-domain and IT-domain training data provided by the IT-translation task organizers. All the three systems domain-adapted: they are trained and tuned on the Batch1 and Batch2 parts of the in-domain training data, as described below. Thus, even without the domain adaptation through in-domain lexicons (which were also provided by the task organizers), the systems constitute strong baselines within the IT domain. Still, the lexicons were not used to train nor tune the systems.

4.1 Moses

Moses is a standard phrase-based statistical machine translation system.

We train Moses on general-domain training data and tune it on the Batch2 part from in-domain training data using MERT (Och, 2003).

We perform domain adaptation of Moses using either XXX placeholders or XML annotations. EN→CS uses factored translation (with part-of-speech tags as additional target-side factors), which is not compatible with the XML annotations, and thus only XXX placeholders are used for EN→CS.

We apply some rather standard pre- and post-processing steps (implemented as Treex blocks).

Preprocessing:

- segmentation into sentences[10]
- tokenization[11,12]
- normalization of quotes, dashes and contracted forms (for EN→CS)[13]
- entity escaping[14]
- truecasing (for EN→NL)[15]/lowercasing

Postprocessing:

- projection of case of identical words from source to target[16]
- sentence capitalization[17]

4.2 TectoMT

TectoMT is a hybrid MT system, combining statistical and rule-based Treex blocks to perform translation with transfer on the layer of tectogrammatical (deep) syntax.

[10]`W2A::ResegmentSentences`
[11]`W2A::TokenizeMoses`
[12]`W2A::TokenizeMorphoDiTa` for EN→CS
[13]`W2W::NormalizeEnglishSentence`
[14]`W2A::EscapeMoses`
[15]`W2A::TruecaseMoses`
[16]`A2A::ProjectCase`
[17]`A2W::CapitalizeSentStart`

We use TectoMT's translation model interpolation (Rosa et al., 2015), uniformly interpolating a translation model trained on the out-of-domain training data with one trained on the Batch1 part of the in-domain training data. Unlike Moses, TectoMT does not support automatic tuning of parameters; however, some parameters were tuned manually using Batch2 from in-domain training data.

We only experiment with domain adaptation of TectoMT via Treex wild attributes (§ 3.4).

4.3 Chimera

Chimera is a system combination of TectoTM and Moses. The input text is first translated by TectoMT, thus creating an additional parallel corpus from the input and the output. This is used to construct a secondary phrase table for Moses, which is then applied to the input to produce the translations (Bojar et al., 2013).

In Chimera, we always use domain adaptation for the TectoMT component (via Treex wild attributes), and only experiment with switching it on or off for the Moses component.

In the experiments reported in this paper, we do not employ the Depfix component of Chimera (Rosa et al., 2012), as it has little relevance to the domain-adaptation problem and would thus clutter the results unnecessarily. However, Depfix is used in the EN→CS Chimera system submitted to the translation task.

4.4 WMT submissions

We submitted the following constrained systems to the IT domain translation task of WMT16:

Chimera with domain adaptation using XXX placeholders in the Moses component. For EN→CS, Depfix is also applied. Also denoted as Chimera-plus.

TectoMT with domain adaptation using Treex annotations. This is the 3rd pilot MT system in the QTLeap project (still in development).

Moses baseline vanilla Moses system, tuned on Batch1 only.[18] QTLeap pilot 0.

Chimera pure non-adapted Chimera system, without Depfix postprocessing. Only submitted for EN→CS.

[18]Unlike the Moses system used in the experiments reported in this paper, which is tuned on the Batch2 portion of the in-domain training data.

System	Annotations	→ES	→NL	→PT
Moses	(not adapted)	22.23	23.40	14.01
	XXX	23.61	24.89	15.47
	XML	**24.22**	**25.41**	**15.58**
Chimera	(not adapted)	26.01	21.82	13.11
	XXX	26.89	**23.52**	14.19
	XML	**27.40**	23.26	**14.21**

Table 1: BLEU evaluation of two forced translation styles for Moses: XXX placeholders and XML markup. For comparison, the non-adapted system is also included.

5 Results and Discussion

We use the WMT16 IT task test set (i.e. Batch3 from the QTLeap corpus[19]) to evaluate our experiments using (case-insensitive) BLEU (Papineni et al., 2002).

First, in Table 1, we compare the two annotation styles we can use for Moses. In general, the XML annotations perform better, in half of the cases leading to a result better by about +0.5 BLEU than that of the XXX placeholders while performing worse only once. Although the documentation in the Moses manual is not very detailed in this respect, we believe that the XML annotations are more palatable to the language model, which can then make meaningful decisions at the boundaries of the force-translated entities, while the XXX placeholders simply constitute out-of-vocabulary items for the language model and thus the context it can use is limited. We therefore stick to XML annotations in Moses in further experiments.[20] Still, even the results obtained using the XXX placeholders are competitive and clearly improve over the non-adapted baseline.

In Table 2, we compare the adapted and non-adapted systems. In all but one case,[21] the domain adapted system performs much better than the non-adapted baseline, with an average gain of +1.3 BLEU.

As for the the individual systems, Moses typ-

[19]http://metashare.metanet4u.eu/go2/qtleapcorpus

[20]However, we used the XXX placeholders in the systems submitted to the WMT16 IT domain translation task, since the XML preprocessing was not implemented in time. Also, for EN→CS we still use the XXX because the XML placeholders are not compatible with the factored translation.

[21] For EN→CS Chimera, our domain adaptation improves case-sensitive BLEU, but worsens case-insensitive BLEU (23.47 vs. 23.36 in Table 2) and also human evaluation (rank 1–2 vs. 4–5 in Table 3). Two possible explanations are: a) the TectoMT component of Chimera is already domain-adapted with gazetteers, b) EN→CS Chimera uses XXX (but for the other languages in Table 2, Chimera uses XML).

System	Adapted	→CS	→ES	→NL	→PT
TectoMT	no	19.98	23.24	18.83	13.87
	yes	**21.89**	**24.31**	**19.89**	**15.51**
Moses	no	23.25	22.23	23.40	14.01
	yes	**23.71**	**24.22**	**25.41**	**15.58**
Chimera	no	**23.47**	26.01	21.82	13.11
	yes	23.36	**27.40**	**23.26**	**14.21**

Table 2: BLEU evaluation of the domain adaptation, using Treex annotations for TectoMT and XML annotations for Moses (except for EN→CS, which uses XXX annotations in Moses).

System	Adapted	→CS	→ES	→NL	→PT
TectoMT	yes	3	**1–2**	3	**1**
Moses	no	4–5	3	4	3
Chimera	no	**1–2**			
	yes	4–5	**1–2**	2	2
another		**1–2**		**1**	

Table 3: Human evaluation ranks of constrained systems in WMT2016 IT-domain task.

ically outperformed both TectoMT and Chimera. The only exception is EN→ES, where TectoMT is stronger than Moses, and the Chimera combination is even stronger than the individual systems.[22]

Table 3 shows results of the human evaluation based on TrueSkill scores (for details, see the overview paper in these proceedings). For EN→CS and EN→ES, there was not better constrained system than Chimera (for EN→CS the non-adapted one, see footnote 21). For EN→NL and EN→PT, Chimera was the second best system (for EN→PT, TectoMT was better than Chimera, in accordance with the BLEU results in Table 2).

Finally, in Table 4, we compare our domain adaptation of Moses (through preprocessing and XML annotations) with the standard approach, where a secondary in-domain phrase table was created from the provided in-domain bilingual lexicons (this experiment was only performed for EN→NL). As could be expected, the standard approach is more powerful, leading to a gain of +4 BLEU, while our approach achieves a +2 BLEU gain. Therefore, the standard approach should be used whenever possible. Still, the fact that our simple and light-weight domain adaptation techniques are able to get half of the achievable improvement is encouraging for scenarios where the standard approach is not applicable.

[22]TectoMT's quality depends not only on the size of training data but also on the taggers, parsers etc. used in the pipeline. Chimera profits from different distribution of errors types in TectoMT and Moses.

Adaptation	→NL
(not adapted)	23.40
XML annotations	25.41
In-domain phrase table	**27.48**

Table 4: BLEU evaluation of the Moses system on EN→NL, comparing the baseline non-adapted Moses, Moses adapted by forced translations annotated with XML markup, and Moses using a secondary in-domain phrase table.

6 Conclusion

Domain adaptation without retraining can be effectively performed through preprocessing and postprocessing, and achieves about half of the quality gain compared to the standard method (training an additional in-domain phrase table), as measured in BLEU improvement above the baseline non-adapted Moses system.

A system-specific forced translation mechanism, such as Moses XML markup, can perform better than simple placeholders. Still, even the placeholders are competitive and may be useful if the MT system in question does not support any mechanism for forcing specific translations.

Acknowledgments

This research was supported by the grants FP7-ICT-2013-10-610516 (QTLeap), GAUK 1572314, SVV 260 333, and using language resources distributed by the LINDAT/CLARIN project of the Ministry of Education, Youth and Sports of the Czech Republic (LM2015071). We thank the two anonymous reviewers for useful comments.

References

Ondřej Bojar, Rudolf Rosa, and Aleš Tamchyna. 2013. Chimera–three heads for English-to-Czech translation. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 90–96.

Hal Daumé III. 2009. Frustratingly easy domain adaptation. *CoRR*, abs/0907.1815.

Rosa Gaudio, Petya Osenova, Kiril Simov, Gorka Labaka, and Dieke Oele. 2016. SMT and hybrid systems of the QTLeap project in the WTM16 IT-task. In *Proceedings of the 1st Conference on Machine Translation*, Stroudsburg, PA, USA. Association for Computational Linguistics.

Philipp Koehn, 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 Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 177–180, Stroudsburg, PA, USA. Association for Computational Linguistics.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 160–167, Stroudsburg, PA, USA. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

Martin Popel and Zdeněk Žabokrtský. 2010. TectoMT: Modular NLP framework. In *Proceedings of the 7th International Conference on Advances in Natural Language Processing*, IceTAL'10, pages 293–304, Berlin, Heidelberg. Springer-Verlag.

Rudolf Rosa, David Mareček, and Ondřej Dušek. 2012. DEPFIX: A system for automatic correction of Czech MT outputs. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, WMT '12, pages 362–368, Stroudsburg, PA, USA. Association for Computational Linguistics.

Rudolf Rosa, Ondřej Dušek, Michal Novák, and Martin Popel. 2015. Translation model interpolation for domain adaptation in TectoMT. In Jan Hajič and António Branco, editors, *Proceedings of the 1st Deep Machine Translation Workshop*, pages 89–96, Praha, Czechia. ÚFAL MFF UK.

Zdeněk Žabokrtský, Jan Ptáček, and Petr Pajas. 2008. TectoMT: Highly modular MT system with tectogrammatics used as transfer layer. In *Proceedings of the Third Workshop on Statistical Machine Translation*, StatMT '08, pages 167–170.