PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 106 OCTOBER 2016 193–204

RuLearn: an Open-source Toolkit for the Automatic Inference of Shallow-transfer Rules for Machine Translation

Víctor M. Sánchez-Cartagena[a], Juan Antonio Pérez-Ortiz[b], Felipe Sánchez-Martínez[b]

[a] Prompsit Language Engineering, Spain
[b] Departament de Llenguatges i Sistemes Informàtics, Universitat d'Alacant, Spain

Abstract

This paper presents ruLearn, an open-source toolkit for the automatic inference of rules for shallow-transfer machine translation from scarce parallel corpora and morphological dictionaries. ruLearn will make rule-based machine translation a very appealing alternative for under-resourced language pairs because it avoids the need for human experts to handcraft transfer rules and requires, in contrast to statistical machine translation, a small amount of parallel corpora (a few hundred parallel sentences proved to be sufficient). The inference algorithm implemented by ruLearn has been recently published by the same authors in *Computer Speech & Language* (volume 32). It is able to produce rules whose translation quality is similar to that obtained by using hand-crafted rules. ruLearn generates rules that are ready for their use in the Apertium platform, although they can be easily adapted to other platforms. When the rules produced by ruLearn are used together with a hybridisation strategy for integrating linguistic resources from shallow-transfer rule-based machine translation into phrase-based statistical machine translation (published by the same authors in *Journal of Artificial Intelligence Research*, volume 55), they help to mitigate data sparseness. This paper also shows how to use ruLearn and describes its implementation.

1. Introduction

Although statistical machine translation (SMT) has been the leading MT paradigm during the last decade, its application may be limited by the availability of parallel corpora. When parallel corpora sufficiently big to build a competitive SMT system are not available, rule-based machine translation (RBMT) is an appealing option. How-

© 2016 PBML. Distributed under CC BY-NC-ND. Corresponding author: vmsanchez@prompsit.com
Cite as: Víctor M. Sánchez-Cartagena, Juan Antonio Pérez-Ortiz, Felipe Sánchez-Martínez. RuLearn: an Open-source Toolkit for the Automatic Inference of Shallow-transfer Rules for Machine Translation. The Prague Bulletin of Mathematical Linguistics No. 106, 2016, pp. 193–204. doi: 10.1515/pralin-2016-0018.

ever, if the RBMT system has to be developed from scratch, the cost in terms of time spent by trained linguists can be prohibitively high.

In this paper, we present ruLearn, an open-source toolkit with which to automatically infer shallow-transfer RBMT rules from very small parallel corpora and existing RBMT dictionaries. The underlying methodology has been described in depth elsewhere (Sánchez-Cartagena et al., 2015): multiple rules with different generalisation levels are generated from bilingual phrases extracted from the parallel corpus and the minimum set of rules that correctly reproduces the bilingual phrases is selected. In this way, conflicts between rules are effectively solved at a global level. The rules produced by ruLearn are encoded in the format used by the Apertium shallow-transfer RBMT platform (Forcada et al., 2011) but they can be adapted to other platforms. They can be easily modified by human experts and can co-exist with hand-crafted rules.

Transfer rules are the linguistic resource in Apertium that requires the deepest linguistic knowledge in order to be created. Apertium translates by analysing the source-language (SL) text into an SL intermediate representation (IR), transferring it into a TL IR, and generating the final translation from the TL IR. The transfer step makes use of transfer rules and bilingual dictionaries while the analysis and generation steps require monolingual morphological dictionaries. Transfer rules encode the operations to be carried out in order to deal with the grammatical divergences between the languages. Thus, ruLearn reduces the difficulty of creating Apertium-based RBMT systems for new language pairs. ruLearn has been successfully used in the development of Apertium-based RBMT systems for Chinese→Spanish (Costa-Jussà and Centelles, 2015) and Serbian↔Croatian (Klubička et al., 2016)

The rules obtained with ruLearn can also be integrated into a phrase-based SMT system by means of the hybridisation strategy we developed (Sánchez-Cartagena et al., 2016) and released as an open-source toolkit (Sánchez-Cartagena et al., 2012). When shallow-transfer rules extracted from the same training corpus are integrated into a phrase-based SMT system, the translation knowledge contained in the parallel corpus is generalised to sequences of words that have not been observed in the corpus, thus helping to mitigate data sparseness.

The rest of the paper is organised as follows: next section presents the most prominent related rule inference approaches in literature. Section 3 describes the rule inference algorithm implemented by ruLearn. A summary of the most relevant results is presented in Section 4. Implementation details and usage instructions are provided in Section 5. The paper ends with some concluding remarks.

2. Related work

There have been other attempts to automatically infer transfer rules for RBMT. ruLearn is greatly inspired by the work of Sánchez-Martínez and Forcada (2009). It overcomes the most relevant limitations of their work: the low expressiveness of their formalism, which is not able to encode rules that are applied regardless of the morphological inflection attributes of the words they match and hence limits the generali-

sation power of their approach;[1] and the fact that their algorithm generates rules that usually prevent the application of other, more convenient rules, when they are used in the Apertium RBMT platform. ruLearn explicitly takes into account the interaction between rules when the RBMT engine chooses which rules to apply and avoids the generation of rules that harm translation quality.

Probst (2005) developed a method with which to learn transfer rules from a small set of bilingual segments obtained by asking bilingual annotators to translate a controlled, parsed corpus. The main differences between her approach and ruLearn are the following: first, her method learns hierarchical syntactic rules that are integrated in a statistical decoder (thus the system can mitigate the impact of errors introduced by the rules) whereas ruLearn produces flat, shallow-transfer rules that are used by a pure RBMT system; and, second, her approach solves conflicts between rules in a greedy fashion rather than choosing the most appropriate ones according to a global minimisation function. Varga and Yokoyama (2009) also developed a rule inference method addressed to small parallel corpora. The differences with ruLearn are similar to those that have just been described: the rules inferred by their approach are also hierarchical syntactic rules that must be used in a system with a statistical decoder.

Finally, Caseli et al. (2006) present a method in which shallow-transfer rules and bilingual dictionaries are learnt from a parallel corpus. It mainly differs from ruLearn in the way in which bilingual phrases are generalised to obtain rules. Unlike ruLearn, their approach does not generalise the rules to unseen values of morphological inflection attributes and deals with conflicts between rules in a greedy manner.

Among the rule inference approaches listed in this section, only those by Sánchez-Martínez and Forcada (2009) and Caseli et al. (2006) have been released as open-source toolkits.[2] We expect ruLearn to be a useful alternative to these tools thanks to its strong generalisation power and its ability to effectively solve rule conflicts.

3. Automatic inference of shallow-transfer rules

3.1. Generalised alignment template formalism

Instead of directly inferring shallow-transfer rules, ruLearn infers simpler units called generalised alignment templates (GATs) which are converted into Apertium shallow-transfer rules at the end of the whole process. GATs are easier to obtain from parallel corpora than Apertium shallow-transfer rules. The SL and TL IRs in Apertium consist of sequences of *lexical forms*. A lexical form, e.g. *car* `N-gen:ϵ.num:sg`, consists of a lemma (*car*), a lexical category (`N` = noun) and a set of morphological inflection attributes and their values (`gen:ϵ.num:sg` = empty gender and singular num-

[1] For instance, four different rules are needed by the approach of Sánchez-Martínez and Forcada (2009) in order to swap a noun followed by an adjective when translating from Spanish to English: one for each possible combination of gender and number.

[2] Available at `https://sourceforge.net/projects/apertium/files/apertium-transfer-tools/` and `https://sourceforge.net/projects/retratos/` respectively.

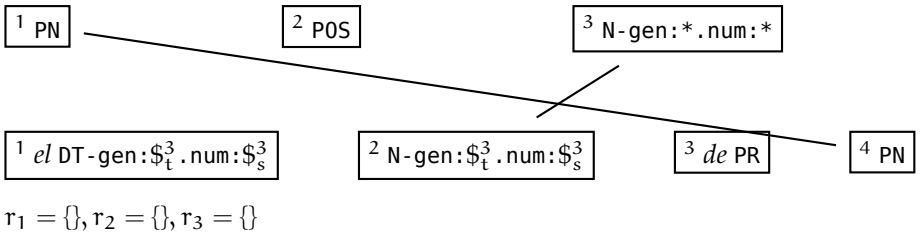

Figure 1. GAT for the translation of the English Saxon genitive construction into Spanish. It will produce the Spanish translation "el gato de Juan" from English "John's cat".

ber). A GAT processes a sequence of SL lexical forms together with their translation according to the bilingual dictionary of the RBMT system and performs the required operations to ensure that the output is grammatically correct in the TL.

Figure 1 shows a GAT that encodes the translation of the English Saxon genitive construction into Spanish. It matches a sequence of 3 English lexical forms defined by the SL word classes depicted at the top of the figure: a proper noun (PN) with any lemma followed by the possessive ending (POS) and a (common) noun with any lemma, any gender and any number. The wildcard value ($*$) for an SL morphological inflection attribute means that any value is allowed. Our formalism also permits defining the lemmas that a sequence of lexical forms must have in order to match a GAT. The GAT in Figure 1 generates a sequence of 4 TL lexical forms defined by the TL word classes: a determiner (DT) whose lemma is *el*, a noun whose lemma is obtained after looking up the SL noun that matched the GAT in the bilingual dictionary (there is an alignment link between them), a preposition (PR) whose lemma is *de* and a proper noun whose lemma is obtained after looking up the SL proper noun in the bilingual dictionary. The genders of the TL determiner and noun are copied from the TL lexical form obtained after looking up the SL noun in the bilingual dictionary ($\$_t^3$; the SL noun is the third matching SL lexical form), while the number is copied from the same SL lexical form without dictionary look-up ($\$_s^3$). Attributes $\$_t^3$ and $\$_s^3$ are *reference attributes* because their values depend on the SL lexical forms that match the GAT. Finally, restrictions (r_i) define the values of morphological inflection attributes the matching SL lexical forms must have after being looked up in the bilingual dictionary in order to match the GAT. In the running example, no restrictions are imposed. See the publication by Sánchez-Cartagena et al. (2015, Sec. 3) for more details.

3.2. Rule inference algorithm

In the first step of the rule inference algorithm implemented by ruLearn (all the steps are summarised in Figure 2), bilingual phrases are obtained from the parallel corpus following a strategy similar to that usually followed for obtaining bilingual phrases during SMT training (Koehn, 2010). From each bilingual phrase, many dif-

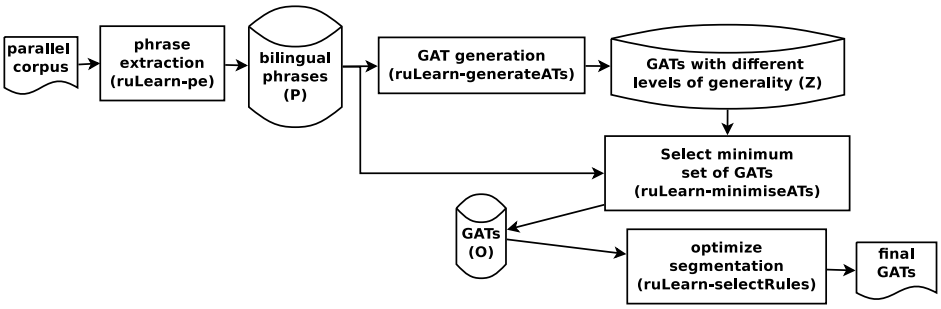

Figure 2. Steps followed to obtain a set of shallow-transfer rules from a parallel corpus.

ferent GATs that correctly *reproduce* it, that is, when applied to the SL phrase, the corresponding TL phrase is obtained, are generated. GATs with different levels of generalisation are obtained by using different sets of wildcard and reference attributes, and also with different lexicalised SL word classes (SL word classes that only match specific lemmas). Only a subset of these GATs will be part of the output of ruLearn.

In order to produce high-quality rules, a set of GATs that correctly translates at least all the bilingual phrases extracted from the parallel corpus should be chosen. These GATs should be as general as possible in order to extend the linguistic knowledge from the corpus to unseen SL segments. This is achieved by selecting the minimum set of GATs needed to correctly reproduce all the bilingual phrases. In addition, just before selecting them, those GATs that correctly reproduce a low proportion of the bilingual phrases they match (the proportion is controlled by a threshold δ) are removed.

The minimisation problem is formalised as follows. Let P be the set of bilingual phrases, Z the set of GATs, $\mathcal{G}(z)$ the set of bilingual phrases correctly reproduced by the GAT $z \in Z$ and $\mathcal{B}(z)$ the set of bilingual phrases matched but not correctly reproduced by z (i.e. when z is applied to the SL side of the bilingual phrase, the TL side is not obtained). The relation of specificity between GATs is defined by the function more_specific(z_i, z_j), whose value is `true` if z_i is more specific than z_j, that is, if z_i contains more lexicalised words or less wildcard and reference attributes than z_j. This function is only defined for GATs with the same sequence of SL lexical categories, as explained later in this section. The minimum set of GATs $O \subseteq Z$ is chosen subject to the following constraints:

1. Each bilingual phrase pair has to be correctly reproduced by at least one GAT that is part of the solution:

$$\bigcup_{z_i \in O} \mathcal{G}(z_i) = P$$

2. If a GAT z_i that is part of the solution incorrectly reproduces a bilingual phrase pair $p \in P$, there is another GAT z_j that is part of the solution, is more specific

than z_i and correctly reproduces p:

$$\forall z_i \in O, \forall p \in \mathcal{B}(z_i), \exists z_j \in O : \text{more\_specific}(z_j, z_i) \ \wedge \ p \in \mathcal{G}(z_j)$$

The solution generally looks like a hierarchy with a mix of general rules and more specific rules fixing the cases not correctly translated with the general ones. The problem can be solved in a reasonable amount of time when the quantity of bilingual phrases and GATs is relatively small (a common situation when the amount of training parallel corpora is scarce) by splitting it into one independent subproblem for each different sequence of SL lexical categories. Each subproblem is formulated as an integer linear programming problem (Garfinkel and Nemhauser, 1972) and solved using the state-of-the-art *branch and cut* algorithm (Xu et al., 2009).

After solving the minimisation subproblems, GATs with certain sequences of SL lexical categories are discarded. This is necessary because, in Apertium, the segmentation of the input SL sentences into chunks (sequences of SL lexical forms that are processed together by a rule) is done by the rules to be applied, which are chosen by the engine in a greedy, left-to-right, longest match fashion. It is necessary to avoid that lexical forms that should be processed together (because they are involved in the same linguistic phenomenon) are assigned to different chunks. The minimum set of SL text segments (*key segments*) in the SL side of the training corpus which need to be translated by a rule to obtain the highest similarity with the TL side is first identified. Afterwards, the set of sequences of SL categories that ensure that the maximum number of key segments get translated properly are selected and those GATs with a sequece of SL lexical categories not found in that set are discarded. Finally, those GATs which produce the same translations that a sequence of shorter GATs would produce are removed and the remaining GATs are encoded as Apertium shallow-transfer rules. More details can be found in the paper by Sánchez-Cartagena et al. (2015, Sec. 4).

4. Evaluation of the tool

The rule inference algorithm implemented by ruLearn was exhaustively evaluated in the paper by Sánchez-Cartagena et al. (2015). Experiments comprised 5 different language pairs. For each of them, shallow-transfer rules were inferred from parallel corpora of different sizes (from 100 to 25 000 parallel sentences) and the resulting rules were integrated in Apertium and automatically evaluated using a test parallel corpus.

The evaluation showed that ruLearn clearly outperforms the approach by Sánchez-Martínez and Forcada (2009). Furthermore, the number of inferred rules is significatly smaller. When the languages involved are closely-related, a few hundred parallel sentences proved to be sufficient to obtain a set of competitive transfer rules, since the addition of more parallel sentences did not result in great translation quality improvements. For instance, Figure 3 shows the results of the automatic evaluation of the Spanish→Catalan rules produced by ruLearn from fragments of differenct sizes

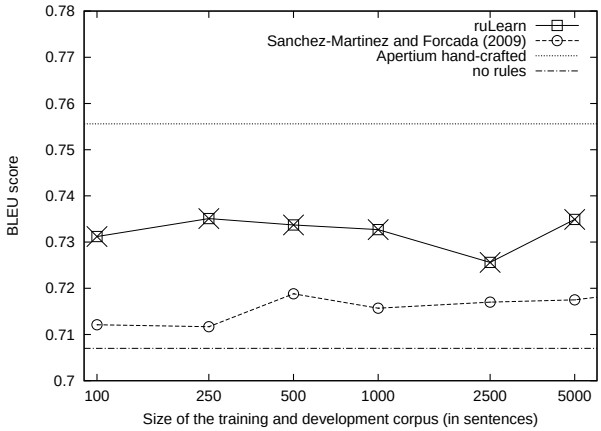

Figure 3. Translation quality (measured using BLEU) achieved by the Spanish→Catalan shallow-transfer rules produced by ruLearn, the rules produced by the approach by Sánchez-Martínez and Forcada (2009), the hand-crafted rules included in Apertium and an empty set of rules. If the difference between the rules obtained with the two rule inference approaches is statistically significant according to paired bootstrap resampling (Koehn, 2004) with $p \leq 0.05$ *and* $1\,000$ *iterations, a diagonal cross is placed on top of the points that represent the results of the approach that performs best.*

of a parallel corpus extracted from the newspaper *El Periodico de Catalunya*.[3] The test corpus was built by randomly selecting sentences from the parallel corpus *Revista Consumer Eroski* (Alcázar, 2005), which contains product reviews. The evaluation metric used was BLEU (Papineni et al., 2002). More details about the evaluation can be found in the paper by Sánchez-Cartagena et al. (2015).

The high complexity of the minimisation problem, which is caused by the generalisation of morphological inflection attributes (with wildcard and reference attributes), made very difficult the evaluation of the inference algorithm with training corpora bigger than $5\,000$ sentences. Disabling that generalisation allowed ruLearn to scale to bigger corpora and reach, and in some cases surpass, the translation quality of the Apertium hand-crafted rules. For instance, Figure 4 shows the results of the automatic evaluation of the Spanish→English rules produced by ruLearn from fragments of different sizes of the *Europarl* (Koehn, 2005) parallel corpus (minutes from the European Parliament). The test corpus was *newstest2013*[4], which contains pieces of news. Note that ruLearn outperforms the hand-crafted rules for the biggest training corpus.

[3] `http://www.elperiodico.com/`

[4] `http://statmt.org/wmt13/translation-task.html`

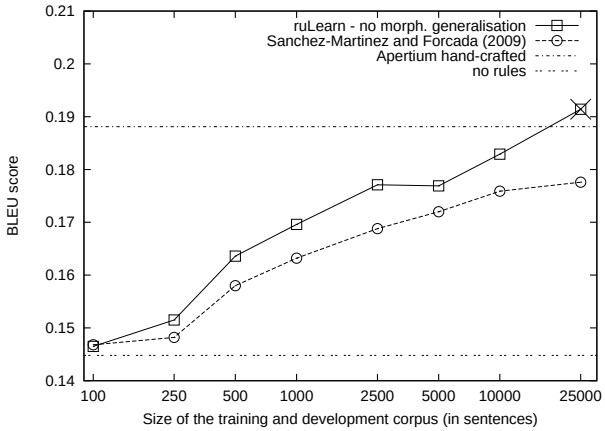

Figure 4. Translation quality (measured using BLEU) achieved by the Spanish→English shallow-transfer rules produced by ruLearn after disabling generalisation of morphological inflection attributes, the rules produced by the approach by Sánchez-Martínez and Forcada (2009), the hand-crafted rules included in Apertium and an empty set of rules. A diagonal cross over a square point indicates that ruLearn outperforms the hand-crafted rules by a statistically significant margin according to paired bootstrap resampling (Koehn, 2004) with $p \leq 0.05$ *and* $1\,000$ *iterations.*

Moreover, we proved that ruLearn can be successfully combined with a hybridisation approach (Sánchez-Cartagena et al., 2016) in order to allow an SMT system enhanced with linguistic information from RBMT to be built using dictionaries as the only hand-crafted linguistic resource. According to our evaluation, a hybrid system with automatically inferred rules is able to attain the translation quality achieved by a hybrid system with hand-crafted rules and, even when it does not, it often performs better than a pure SMT system and a hybrid system that only uses RBMT dictionaries.

5. Technical details and usage instructions

5.1. Getting ruLearn

ruLearn source code can be downloaded from the Apertium Subversion repository at `https://svn.code.sf.net/p/apertium/svn/trunk/ruLearn`. It is licensed under GNU GPL v3. and distributed as a GNU Autotools[5] package. It currently can only be compiled and executed under GNU/Linux.

[5] `http://www.gnu.org/software/autoconf/` and `http://www.gnu.org/software/automake/`

```
<np> <pos> <n><*gen><*num> |
el<det><def><)3gen><(3num> <n><)3gen><(3num> de<pr>
<np> | 0:3 1:0 2:1 |
```

Figure 5. GAT in Figure 1 encoded in one the intermediate files generated by ruLearn. Fields are separated by |. The first field represents SL word classes, the second field contains TL word classes, the third one contains word alignments and the last one, restrictions.)3 represents the reference attribute $\$_t^3$ in Figure 1, while (3 represents $\$_s^3$.

5.2. Program design

ruLearn is written in `Bash` and `Python`. There is an independent command-line program for each step of the algorithm (their names are depicted in Figure 2) and a wrapper program that executes all the steps. Given the huge amount of bilingual phrases and GATs that need to be processed, communication between the different modules is done by writing and reading intermediate files. The results of each step of the algorithm are written in a different subdirectory. This allows users to understand the different steps of the algorithm and even to customise the algorithm by adding new steps. It also makes easier the reuse of some of the steps from previous executions of the algorithm. `Bash` is used to manage and check the availability of all the intermediate files while the core algorithm is implemented in `Python`.

GATs for each sequence of SL lexical categories are stored in a different file (Figure 5 shows how the GAT in Figure 1 is encoded in an intermediate file) and bilingual phrases are organised in a similar way. This way of storing the data eases the parallelisation of the different minimisation subproblems and increases the simplicity of the core algorithm code. By default, all the available CPUs of the machine are used to solve the minimisation subproblems thanks to the use of the `parallel` tool.[6] In order to increase the parallelisation degree and hence speed up the process, the minimisation subproblems can be scattered across different machines.

5.3. Usage instructions

Compilation and installation of ruLearn can be performed by means of the commands depicted below. The `configure` program checks whether all the software dependencies are met. The most important ones are a recent version of Apertium and the `PuLP`[7] Python module, which contains the linear programming solver.

```
$ ./autogen.sh
$ ./configure && make && make install
```

[6] `https://joeyh.name/code/moreutils/`

[7] `http://pypi.python.org/pypi/PuLP`

```
[tag groups]
gender:m,f,mf,GD,nt
number:sg,pl,sp,ND
...
[tag sequences]
n:gender,number
...
```

Figure 6. Fragment of a linguistic configuration file. The `[tag groups]` section defines the values the morphological inflection attributes can take. The `[tag sequences]` section defines that all the nouns must contain a gender and a number.

In order to produce rules, ruLearn needs a training parallel corpus, a development corpus (used to determine the best value for the threshold δ described in Section 3.2), the path to the source code of the Apertium linguistic package of the language pair for which rules will be inferred (because Apertium linguistic resources are needed in order to analyse the training corpus) and a linguistic configuration file, which contains a set of Apertium-specific and language-pair-dependent configuration parameters. The most important ones are *tag groups* and *tag sequences*. The former define the allowed values for each type of morphological inflection attribute while the latter define the sequence of attributes for each lexical category (Figure 6 shows an example). They are needed in order to map lexical forms as they encoded in the Apertium dictionaries to a representation compatible with the GAT formalism, in which the type of each morphological inflection attribute is explicitly defined. For instance, a feminine singular noun is represented in Apertium as `<n><f><sg>` (`f` stands for *feminine* and `sg` stands for *singular*). The fact that `f` represents a gender and the set of possible values a gender can take is not explitly encoded anywhere in Apertium, but this information is needed by the rule inference algorithm in order to be able to introduce wildcard and reference attributes. Examples of linguistic configuration files for different language pairs are shipped with ruLearn.

The following command runs the rule inference algorithm:

```
$ ruLearn --source_language SOURCE_LANGUAGE_CODE --target_language
TARGET_LANGUAGE_CODE  --corpus TRAINING_CORPUS --dev_corpus
DEVELOPMENT_CORPUS --data_dir SOURCE_OF_APERTIUM_LANGUAGE_PAIR
--work_dir OUTPUT_DIRECTORY --config LINGUISTIC_CONFIG_FILE
```

Results are written into the directory `OUTPUT_DIRECTORY`. When the inference process finishes, ruLearn prints the best value of δ and the path to the file with the resulting set of rules. If a test corpus is defined with the `--test_corpus` option, ruLearn translates it with the automatically inferred rules and prints the BLEU and TER scores obtained.

6. Concluding remarks

We have presented ruLearn: an open-source toolkit for the automatic inference of shallow-transfer rules from scarce parallel corpora and morphological dictionaries. ruLearn produces rules that can be used in the Apertium platform without further modification and are able to reach the translation quality of hand-crafted rules. The software architecture of the toolkit allows it to deal with the complexity of the rule inference algorithm by introducing a high degree of parallelisation.

Concerning future research lines, the rule formalism could be extended with a new type of GAT in order to further improve the generalisation power and the translation quality achieved between languages that are not closely related. These new GATs would operate on sequences of chunks instead of sequences of words and would be encoded as Apertium *interchunk* rules (Forcada et al., 2011, Sec. 2.1). ruLearn could also be used when a parallel corpus is not available if a crowdsourcing (Wang et al., 2013) approach is followed. Finally, we plan to integrate our open-source tool for hybridisation (Sánchez-Cartagena et al., 2012) into ruLearn in order to ease the use of automatically inferred rules in a phrase-based SMT system.

Acknowledgements

Research funded by the Spanish Ministry of Economy and Competitiveness through projects TIN2009-14009-C02-01 and TIN2012-32615, by Generalitat Valenciana through grant ACIF/2010/174, and by the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement PIAP-GA-2012-324414 (Abu-MaTran).

Bibliography

Alcázar, A. Consumer Corpus: Towards linguistically searchable text. In *Proceedings of BIDE (Bilbao-Deusto) Summer School of Linguistics 2005*, Bilbao, Spain, 2005.

Caseli, H. M., M. G. V. Nunes, and M. L. Forcada. Automatic induction of bilingual resources from aligned parallel corpora: application to shallow-transfer machine translation. *Machine Translation*, 20(4):227–245, 2006.

Costa-Jussà, M. R. and J. Centelles. Description of the Chinese-to-Spanish Rule-Based Machine Translation System Developed Using a Hybrid Combination of Human Annotation and Statistical Techniques. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 15(1), 2015.

Forcada, M. L., M. Ginestí-Rosell, J. Nordfalk, J. O'Regan, S. Ortiz-Rojas, J. A. Pérez-Ortiz, F. Sánchez-Martínez, G. Ramírez-Sánchez, and F. M. Tyers. Apertium: a free/open-source platform for rule-based machine translation. *Machine Translation*, 25(2):127–144, 2011. Special Issue: Free/Open-Source Machine Translation.

Garfinkel, R. S. and G. L. Nemhauser. *Integer programming*, volume 4. Wiley New York, 1972.

Klubička, F., G. Ramírez-Sánchez, and N. Ljubešić. Collaborative development of a rule-based machine translator between Croatian and Serbian. *Baltic Journal of Modern Computing*, 4(2), 2016.

Koehn, P. Statistical significance tests for machine translation evaluation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, volume 4, pages 388–395, Barcelona, Spain, 2004.

Koehn, P. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of the Machine Translation Summit X*, pages 12–16, Phuket, Thailand, September 2005.

Koehn, P. *Statistical Machine Translation*. Cambridge University Press, 2010.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. doi: 10.3115/1073083.1073135. URL `http://www.aclweb.org/anthology/P02-1040`.

Probst, K. *Automatically Induced Syntactic Transfer Rules for Machine Translation under a Very Limited Data Scenario*. PhD thesis, Carnegie Mellon University, 2005.

Sánchez-Cartagena, V. M., F. Sánchez-Martínez, and J. A. Pérez-Ortiz. An open-source toolkit for integrating shallow-transfer rules into phrase-based satistical machine translation. In *Proceedings of the Third International Workshop on Free/Open-Source Rule-Based Machine Translation*, pages 41–54, Gothenburg, Sweden, June 2012.

Sánchez-Cartagena, V. M., J. A. Pérez-Ortiz, and F. Sánchez-Martínez. A generalised alignment template formalism and its application to the inference of shallow-transfer machine translation rules from scarce bilingual corpora. *Computer Speech & Language*, 32(1):46–90, 2015. Hybrid Machine Translation: integration of linguistics and statistics.

Sánchez-Cartagena, V. M., J. A. Pérez-Ortiz, and F. Sánchez-Martínez. Integrating rules and dictionaries from shallow-transfer machine translation into phrase-based statistical machine translation. *Journal of Artificial Intelligence Research*, 55:17–61, 2016.

Sánchez-Martínez, F. and M. L. Forcada. Inferring shallow-transfer machine translation rules from small parallel corpora. *Journal of Artificial Intelligence Research*, 34(1):605–635, 2009.

Varga, I. and S. Yokoyama. Transfer rule generation for a Japanese-Hungarian machine translation system. In *Proceedings of the Machine Translation Summit XII*, Ottawa, Canada, 2009.

Wang, A., C. Hoang, and M.Y. Kan. Perspectives on crowdsourcing annotations for natural language processing. *Language Resources and Evaluation*, 47(1):9–31, 2013.

Xu, Y., T. K. Ralphs, L. Ladányi, and M. J. Saltzman. Computational experience with a software framework for parallel integer programming. *INFORMS Journal on Computing*, 21(3), 2009.

Address for correspondence:
Víctor M. Sánchez-Cartagena
`vmsanchez@prompsit.com`
Prompsit Language Engineering
Av. Universitat s/n. Edifici Quorum III. E-03202 Elx, Spain