PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 100 OCTOBER 2013 73–82

XenC: An Open-Source Tool for Data Selection in Natural Language Processing

Anthony Rousseau

Laboratoire d'Informatique de l'Université du Maine (LIUM)

Abstract

In this paper we describe XenC, an open-source tool for data selection aimed at Natural Language Processing (NLP) in general and Statistical Machine Translation (SMT) or Automatic Speech Recognition (ASR) in particular. Usually, when building a SMT or ASR system, the considered task is related to a specific domain of application, like news articles or scientific talks for instance. The goal of XenC is to allow selection of relevant data regarding the considered task, which will be used to build the statistical models for such a system. It is done by computing the difference between cross-entropy scores of sentences from a large out-of-domain corpus and sentences from a corpus considered as in-domain for the task. Written in C++, this tool can operate on monolingual or bilingual data and is language-independent. XenC, now part of the LIUM toolchain for SMT, is actively developed since December 2011 and used in many MT projects.

1. Introduction

In Natural Language Processing, in general, and in Statistical Machine Translation or Automatic Speech Recognition, in particular, a system and its models are often considered as dynamic, always-evolving entities. These statistical models are not usually set in stone since they can be adapted to a target task or re-estimated with new or additional data. Also, their performance can be enhanced by various techniques, which can occur before, during or after the actual system processing. Among these, one of the most efficient pre-processing technique is data selection, i.e. the fact to carefully choose which data will be injected into the system we are going to build.

In this paper, while focusing on the Statistical Machine Translation field, we describe an open-source tool named XenC, which can be used to easily perform a data

 Corresponding author: anthony.rousseau@lium.univ-lemans.fr
Cite as: Anthony Rousseau. XenC: An Open-Source Tool for Data Selection in Natural Language Processing. The Prague Bulletin of Mathematical Linguistics No. 100, 2013, pp. 73–82. doi: 10.2478/pralin-2013-0013.

selection for both monolingual data, aimed at Language Models (LM), and bilingual data, aimed at Translation Models (TM). This tool is freely available for both commercial and non-commercial use and is released under the GNU General Public License version 3.[1] Its most recent source code is accessible at: `https://github.com/rousseau-lium/XenC`.

The paper is organized as follows: in Section 2, we expose the motivations of our work on this tool. Section 3 describes the tool and the way it works. In Section 4, we present the requirements for the usage of the tool. Section 5 is dedicated to usage instructions, i.e. how to run `XenC` efficiently. In Section 6 we present some experimental results to illustrate the interest of such a tool. Then, Section 7 concludes this paper and expose some future plans for `XenC`.

2. Motivations

Most of the time, a translation system is built to fit a given task or a specific domain of application, like medical reports or court session transcriptions. This implies to dispose of a suitable corpus, which can be viewed as *in-domain*, reasonably large to produce an efficient system. Unfortunately, this is rarely the case, as most of the corpora sets usually available in SMT are quite generic and large quantities of relevant data for a desired task or domain are generally difficult to find. These corpora, not adapted for a particular task, can be viewed as *out-of-domain*. Moreover, another issue arising from using such generic corpora is that they can contain useless, or worse, harmful data for the models we want to estimate, thus lowering the translation quality.

With this in mind, the main idea behind `XenC` is to allow the extraction of relevant sentences (regarding the target translation task or domain) from an *out-of-domain corpus* by comparing them to the sentences of an *in-domain corpus*. Based on previous theoretical work by Moore and Lewis (2010) for monolingual selection and Axelrod et al. (2011) for bilingual selection, `XenC` uses cross-entropy (the average negative log of a sentence LM probabilities) as a metric to evaluate and sort those sentences.

Another motivation for our work on `XenC` is that a typical trend in SMT is to use as much data as possible to build statistical models, as long as this growing amount of data will provide a better BLEU score or any other translation quality automatic measure. However, the drawback of this trend is that the size of the models increases very quickly and become much more resource-demanding. So, in order to build either easily deployable systems or to estimate models on limited physical resources, it seems essential to consider resource usage like memory and computation time, both for models estimation and decoding process. Obviously, building a small system with very few data to attain this objective is quite trivial, but it often leads to important translation quality losses, so the goal of `XenC` is to provide a mean to extract small

[1] `http://www.gnu.org/licenses/gpl.html`

amounts of data, carefully selected to match the desired translation task. This way, small but efficient systems can be built. Most of the time, performance of such systems will be better than a system built from all available but generic data, in terms of translation quality, memory usage and computation time.

3. Tool Description

XenC is a tool written in C++, which possesses four filtering modes. The common framework of all these modes is, from an *in-domain* corpus and one or several *out-of-domain* corpora, to first estimate two language models. Currently, all the LM estimations are handled by calls to the SRILM toolkit (Stolcke, 2002) libraries. These two models will then be used to compute two scores for each sentence of the *out-of-domain* corpus so the difference between these scores will provide an estimation of the closeness of each sentence regarding the considered task. In the remainder of this section, we will describe the modes and other functionalities proposed by XenC.

3.1. Processing Modes

The first mode is a filtering process based on a simple perplexity computation, as described in Gao et al. (2002). This is the simplest filtering mode proposed by XenC. Although it can provide interesting results and is less resource-demanding than the other modes, it is also less efficient.

The second mode is based on the monolingual cross-entropy difference as proposed by Moore and Lewis (2010). The cross-entropy is mathematically defined as:

$$H(P_{LM}) = -\frac{1}{n}\sum_{i=1}^{n} \log P_{LM}(w_i|w_1, \cdots, w_{i-1}) \quad (1)$$

where P_{LM} is the probability of a LM for the word sequence W and $w_1, \cdots, w_{k-1}$ represents the history of the word w_i. In this mode, the first LM is estimated from the whole *in-domain* corpus. The second LM is estimated from a random subset of the *out-of-domain* corpus, with a number of tokens similar to the *in-domain* one. Formally, let I be our *in-domain* corpus and N our *out-of-domain* one. $H_I(s)$ will be the cross-entropy of a sentence s of N given by the LM estimated from I, while $H_N(s)$ will be the cross-entropy of sentence s of N given by the LM estimated from the subset of N. The sentences $s_1, \cdots, s_N$ from the *out-of-domain* corpus N will then be evaluated by $H_I(s) - H_N(s)$ and sorted by their score. Although this is a monolingual selection, this mode can be used efficiently on both monolingual and bilingual data.

The third mode is based on the bilingual cross-entropy difference as described in Axelrod et al. (2011). Unlike the second mode, we now take into account the two languages in our computations. Formally, let I_S and I_T be our *in-domain* corpus in source S and target T languages, and N_S and N_T our *out-of-domain* corpus with the same

language pair. For each language, we first compute the monolingual cross-entropy difference as described in the preceding paragraph. The final score will the be computed by the sum between the two cross-entropy differences, as shown in the following equation:

$$[H_{I_S}(s_S) - H_{N_S}(s_S)] + [H_{I_T}(s_T) - H_{N_T}(s_T)] \quad (2)$$

where s_S is a word sequence from the *out-of-domain* corpus in source language and s_T is the corresponding word sequence from the *out-of-domain* corpus in target language.

The last mode operates similarly to the third one, but uses two phrase tables from the Moses toolkit (Koehn et al., 2007) as an input. Its goal is to adapt a phrase table considered as *out-of-domain* with another smaller phrase table considered as *in-domain*. First, source and target phrases are extracted from the phrase tables. Then, just like the third mode, LMs are estimated and used to score each *out-of-domain* phrase in each language. Finally, the scores are inserted in the original phrase table as a sixth feature. Another option is to compute local scores, relative to each unique source phrase. The redundant source phrases are merged into one structure containing their related target phrases, then the scores are computed locally and can be inserted in the original phrase table as a seventh feature. These two new features can then be added to the Moses configuration file for the *out-of-domain* translation system, and their weights tuned along with the other weights. Please note that this fourth mode is currently experimental and is barely tested.

3.2. Other Functionalities

Since the beginning of the `XenC` development right after the IWSLT 2011 evaluation campaign, back in December 2011, three main functionalities have been developed around the filtering modes to enhance them.

The first functionality added to `XenC` comes from an observation we made concerning the strong relation between the selected sentences and the random subset from the *out-of-domain* corpus. Indeed, the scores can vary significantly from one sample to another, impacting the resulting selection. Thus, we implemented a way to reduce this impact by optionally allowing to extract three random subsets instead of one for LM estimation. With this option, for each sentence to score, a cross-entropy measure is computed from each of the three language models. The three scores are then interpolated and used to compute the usual cross-entropy difference as described before. Our experiments shown that this option most of the time leads to a better selection than with only one random subset. It can be used within both the monolingual and bilingual cross-entropy filtering modes.

Our second added functionality is an option to perform the whole (monolingual or bilingual) filtering process on stemmed *in-domain* and *out-of-domain* corpora corresponding to the textual ones. These stemmed corpora must be created with an external tool. For this task, we recommend the TreeTagger tool (Schmid, 1995) which is efficient and language-independent. In order to ease the process of stemming the

corpora, a wrapper script exists within the Moses toolkit. Once the stemmed corpora are generated, distinct LMs and scores will be computed, then these scores will be merged with the ones from the original text corpora. Although this option is still experimental at the time of writing and has been barely tested, our initial experiments showed that an improvement can be achieved, and that integrating stems into the process can lead to a more heterogeneous selection, thus preventing the risk of increasing the number of out-of-vocabulary tokens (OOVs) in the resulting translation system. Again, this option is available for both the monolingual and bilingual filtering modes.

The third and last functionality implemented into `XenC` is the computation of cosine similarity measures in addition to the usual cross-entropy scores. In Information Retrieval, this measure is used for document clustering where each document is represented by a vector and vectors are compared by computing the cosine of the angle between them. By first determining a common vector of words, then considering the *in-domain* corpus as one document and each *out-of-domain* sentence as documents too, it is possible to obtain similarity scores for each sentence of the said corpus. Currently, `XenC` proposes two options regarding this similarity measure. It is possible to either combine this score with the cross-entropy one or to use it as a stand-alone selection criterion. Since this option has been added very recently, it is still highly experimental and needs extensive testing. To this date, no real improvements have been observed. Also, please note that this option is only available within the monolingual filtering mode.

Some other scoring options are available to fit different scoring needs. For instance, you can provide `XenC` a file containing weights for each sentence of the *out-of-domain* corpus. These weights can optionally be used as log values. Also, you can require a descending sorting order for you final scored file, which can prove useful when you need `XenC` to adapt to some existing scripts. Finally, by default, `XenC` proposes calibrated scores ranging from 0 (the best score) to 1 (the worst one). You can require our tool to invert those scores and have 1 being the best score and 0 the worst one.

4. Installation Requirements

In order in compile and install `XenC` from the source code right out-of-the-box, you will need a Linux (i386 or x86_64), Mac OSX (Darwin) or SunOS (Sparc or i386) operating system. Other platforms may work, but are totally untested. Also, you will need to dispose of the following third-party software:

- `gcc` version 4.2.1 or higher (older versions might work, but are untested),
- GNU `make`,
- `gzip`, to read/write compressed files,

- Boost[2] version 1.52.0 or higher (although it may work with lower versions, but is untested). XenC relies on the following multithreaded ("-mt" versions) libraries: filesystem, iostreams, program_options, regex, system and thread,
- SRILM[3] version 1.7.0 or higher (older versions won't work for sure, since they are not thread-safe). XenC relies on the following libraries: libdstruct, libmisc and liboolm.

Once all third-party software is installed, you can simply compile XenC by issuing the following command: `make`, or `make debug` if you want to keep the debug symbols. You can also specify custom paths for Boost or SRILM, by adding the `BOOST=` or `SRILM=` parameters.

5. Usage Instructions

By default, in order to run XenC, you need to provide at least:

- a source (and optionally target) language,
- an *in-domain* monolingual or parallel corpus,
- an *out-of-domain* monolingual or parallel corpus,
- a filtering mode.

The tool will then compute the *out-of-domain* sentences scores, generating all the needed vocabularies and language models when appropriate, and will output an ascending order sorted file (compressed with `gzip`), containing the scores in the first field and the sentences in the second (and third in case of parallel corpora) field(s). It is mandatory that the original corpora files do not contain tabulations. Empty lines are not an issue, since XenC will automatically skip them and also remove the corresponding sentences in the case of a parallel corpus. Automatic generation of needed files works as follows:

- for vocabularies, the words contained in the *in-domain* corpus will be used,
- for language models, estimation will be done using an order of four, a modified Kneser-Ney discounting and no cut-offs. LMs will be outputted in SRILM binary format.

You can of course provide your own vocabularies and LMs, and you can optionally change the order and the output format of the estimated LMs.

Concerning the evaluation process, it is based on perplexity computation of language models estimated from parts of various sizes of the sorted output file. Concretely, XenC will extract cumulative parts based on a fixed step size (usually ten percent), estimate language models on them, and then compute their perplexity against a development corpus. Our tool also propose a best point computation, which, from the evaluation mode perplexity distribution, will try to find the best percentage of the *out-of-domain* corpus to keep, based on a dichotomic search.

[2] http://www.boost.org

[3] http://www.speech.sri.com/projects/srilm/download.html

Regarding the performance, some parts of our tool are threaded, like the perplexity and cross-entropy computation (since the sentence order does not matter) as well as the language models estimation when evaluating. By default, `XenC` makes use of two threads, and we have successfully ran it with up to ten threads. But due to some memory leaks in the SRILM toolkit, the memory usage can become very important during the evaluation process. It is possible to limit this memory usage by requiring less threads, or by launching `XenC` twice, once for the selection process and once for the evaluation, instead of once for the whole procedure.

5.1. Usage Examples

The simplest command line which can be issued could be the following:

```
XenC -s fr -i indomain.fr -o outofdomain.fr -m 2 --mono
```

where `-s` indicates the source language, `-i` the *in-domain* corpus, `-o` the *out-of-domain* corpus, `-m` the filtering mode and `--mono` forces monolingual mode.

The following line:

```
XenC -s fr -i indomain.fr -o outofdomain.fr -m 2 --mono -e -d dev.fr
```

adds the evaluation mode (the `-e` switch) and `-d` provides the development corpus. To require best point computation, just replace the `-e` switch with the `-b` one.

The last example computes a bilingual filtering with a best point computation and eight threads:

```
XenC -s en -t fr -i indomain.en -o outofdomain.en -d dev.en \
--in-ttext indomain.fr --out-ttext outofdomain.fr -m 3 -b --threads 8
```

Please note that for now, the evaluation or best point can only be done on source language.

6. Experiments

We have performed a series of experiments based on the system we proposed for the IWSLT 2011 evaluation campaign, which achieved the first place in the speech translation task (Rousseau et al., 2011). This system was already based on a very basic perplexity data selection, which explain the fact that size reductions are not reported for the translation tables. We will present our results on selection for language modeling and translation modeling. For these selections, we consider the TED corpus as our *in-domain* one and all the other allowed corpora as our *out-of-domain* ones. The development and test corpora are the official sets proposed during the IWSLT 2010 campaign. Source language is English while target language is French. More detailed experiments can be found in Chapter 6 of Rousseau (2012).

6.1. Data Selection for Language Modeling

The original LM that we used for the evaluation campaign was estimated on all the available data, using a linear interpolation. To study the impact of the monolingual

Systems	dev2010 BLEU	tst2010 BLEU	LM size	
			On disk	In memory
IWSLT11 original	23.97	25.01	7.9G	22.1G
IWSLT11 XenC_LM	**24.01**	**25.35**	1.7G	5.2G

Table 1. BLEU scores and LM sizes with both original and reduced LMs.

Systems	dev2010	tst2010
IWSLT11 original	23.97	25.01
IWSLT11 XenC_monoEN	**24.11**	**25.12**
IWSLT11 XenC_monoFR	24.01	24.87
IWSLT11 XenC_biENFR	**24.10**	**25.13**

Table 2. BLEU scores for bilingual selection for translation models.

data selection, we performed it on each of the *out-of-domain* corpora and interpolated the resulting LMs linearly to create a new reduced LM. We ended up keeping only 11.3% of the original data according to the best point computation of `XenC`. Table 1 presents the BLEU scores obtained by our system for both the original LM and the reduced one, as well as the sizes of the two language models on disk and in memory. As we can observe, our reduced language model achieves better results that the original one, while requiring much less memory and disk space, thus also optimizing the decoding time and memory usage.

6.2. Data Selection for Translation Modeling

We also studied the impact of bilingual selection on all the *out-of-domain* corpora used for the translation model estimation. We made three different selections to compare the efficiency of bilingual selection to monolingual selection on both source and target sides. Table 2 shows the results obtained for each of these selections. As we can see, monolingual source selection and bilingual selection also achieve better results than the original system, while monolingual target selection reduce the translation quality and is therefore not suitable for translation models estimation.

6.3. Data Selection for the Whole System

After studying the individual impact of both monolingual and bilingual data selection, we combined the reduced models to observe if it is possible to achieve even better results than individual selections. Table 3 details the results obtained by the global systems for both monolingual source and bilingual selection. We can observe

Systems	dev2010	tst2010
IWSLT11 original	23.97	25.01
IWSLT11 XenC monoEN + LM	24.12	25.18
IWSLT11 XenC biENFR + LM	**24.18**	**25.40**

Table 3. BLEU scores for the complete experimental systems.

that although source monolingual and bilingual data selection results for the translation model were very similar when performed individually, we can achieve much better results with bilingual selection when the reduced language model is added to the system. In the end, we can report on this particular task a gain of 0.21 BLEU point on the development set and 0.39 BLEU point on the test set, which represents respectively a relative gain of 0.87% and 1.54%.

7. Conclusion and Perspectives

In this paper, we described `XenC`, an open-source tool for data selection in Natural Language Processing. While focusing our experiments on Statistical Machine Translation, we showed that with the help of our tool, carefully selecting the data injected in the building process of translation and language models dedicated to a specific task might lead to smaller models, reduced decoding time and better translation quality.

In the future, we plan to keep the tool development active, as we already have some improvements in mind:

- integrating other language model toolkits and particularly KenLM (Heafield, 2011) for speed and memory usage,
- proposing an option to use the full vocabulary of the two corpora, as it might lead to a reduced OOVs rate,
- extensively testing and enhancing the experimental functionalities,
- proposing an option to evaluate on the target language when doing bilingual selection.

Bibliography

Axelrod, Amittai, Xiaodong He, and Jianfeng Gao. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 355–362, July 2011.

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

Heafield, Kenneth. KenLM: faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197, July 2011.

Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open source toolkit for statistical machine translation. In *Meeting of the Association for Computational Linguistics*, pages 177–180, 2007.

Moore, Robert C. and William Lewis. Intelligent selection of language model training data. In *Proceedings of the ACL Conference Short Papers*, pages 220–224, July 2010.

Rousseau, Anthony. *La Traduction Automatique De La Parole*. PhD thesis, Université du Maine, December 2012.

Rousseau, Anthony, Fethi Bougares, Paul Deléglise, Holger Schwenk, and Yannick Estève. LIUM's systems for the IWSLT 2011 speech translation tasks. In *Proceedings of International Workshop on Spoken Language Translation*, pages 79–85, December 2011.

Schmid, Helmut. Improvements in part-of-speech tagging with an application to German. In *Proceedings of the ACL SIGDAT-Workshop*, pages 47–50, 1995.

Stolcke, Andreas. SRILM – an extensible language modeling toolkit. In *Proceedings of Interspeech*, pages 901–904, September 2002.

Address for correspondence:
Anthony Rousseau
anthony.rousseau@lium.univ-lemans.fr
Laboratoire d'Informatique de l'Université du Maine (LIUM)
Avenue Laënnec
72085 LE MANS CEDEX 9, France