LetsMT!: Cloud-Based Platform for Building User Tailored Machine Translation Engines

Andrejs Vasiļjevs
Tilde
Vienibas gatve 75a, Riga
LV1004, LATVIA
andrejs@tilde.com

Raivis Skadiņš
Tilde
Vienibas gatve 75a, Riga
LV1004, LATVIA
raivis.skadins@tilde.com

Jörg Tiedemann
Uppsala University
Box 635, Uppsala
SE-75126, SWEDEN
jorg.tiedemann@lingfil.uu.se

Abstract

To fully exploit the huge potential of existing open SMT technologies and user-provided content, we have created an innovative online platform for data sharing and MT building. This platform is being developed in the EU collaboration project LetsMT!. This paper presents motivation in developing this platform, its architecture and main features.

1 Introduction

The goal of the LetsMT! project is to facilitate the use of open source SMT toolkits and to involve users in the collection of training data. This will result in populating and enhancing the currently most progressive MT technology and making it available and accessible for all categories of users in the form of sharing MT training data and building tailored MT systems for different languages on the basis of the online LetsMT! platform. The LetsMT! project extends the use of existing state-of-the-art SMT methods enabling users to participate in data collection and MT customization to increase quality, scope and language coverage of MT. Currently LetsMT! is creating a cloud-based platform that gathers public and user-provided MT training data and generates multiple MT systems by combining and prioritizing this data.

The LetsMT! Consortium includes the project coordinator Tilde, the Universities of Edinburgh, Zagreb, Copenhagen and Uppsala, the localization company Moravia and the semantic technology company SemLab. The project started in March 2010 and should achieve its goals till September 2012.

2 Applying user-provided data for SMT training

The number of open source parallel resources is limited and this is an essential problem for SMT, since translation systems trained on data from a particular domain, e.g. parliamentary proceedings, will perform poorly when used to translate texts from a different domain, e.g. news articles. At the same time, a huge amount of parallel texts and translated documents are at the users' disposal and they can be used for SMT system training. Therefore, the LetsMT! online platform provides all categories of users (public organizations, private companies, individuals) with an opportunity to upload their proprietary resources to the repository and to receive a tailored SMT system trained on these resources. The latter can be shared with other users who can exploit them further on. Data and SMT model sharing can be managed by the users. In LetsMT! we emphasize data integrity and security that makes it possible to work with proprietary collections as well as public sources.

As opposed to the data sharing approach by TAUS Data Association[1], LetsMT! allows only to upload data and use it for the training of MT engines. It does not allow to download stored data.

The motivation of users to get involved in sharing their resources is based on the following factors:

- participate and contribute in a reciprocal manner with a community of professionals and its goals;

[1] http://www.tausdata.org/

- achieve better MT quality for user specific texts;
- build tailored and domain specific translation services;
- enhance reputation for individuals and businesses;
- ensure compliance with the requirement set forth by EU Directive to provide usability of public information in a convenient way for public institutions;
- deliver a ready resource for study and teaching purposes for academic institutions.

The LetsMT! project is advancing the concept of data sharing, which implies the practice of making data used in one activity available to other users.

LetsMT! platform provides the following key features:

- Uploading of parallel texts for users that will contribute their content;
- Directory of web and offline resources gathered by LetsMT! users;
- Automated training of SMT systems from specified collections of training data;
- Custom building of MT engines from selected pools of training data;
- Custom building of MT engines from proprietary non-public data;
- MT evaluation facilities.

3 Architecture overview

Figure 1 illustrates the general architecture of the LetsMT! platform. Its components for SMT training, parallel data collection and data processing are described further down in this paper. The development of the system was particularly facilitated by the open-source alignment tool GIZA++ (Och et al. 2002) and the MT training and decoding tool Moses (Koehn et al 2007).

LetsMT! translation services can be used in several ways: through the web portal, through a widget provided for free inclusion in a web-page, through browser plug-ins, and through integration in computer-assisted translation (CAT) tools and different online and offline applications. Localisation and translation businesses as well as other professional translation can use the LetsMT! platform for uploading their parallel corpora in the LetsMT! website, building custom SMT solutions from the specified collections of training data, and accessing these solutions in their productivity environments (typically, various CAT tools).

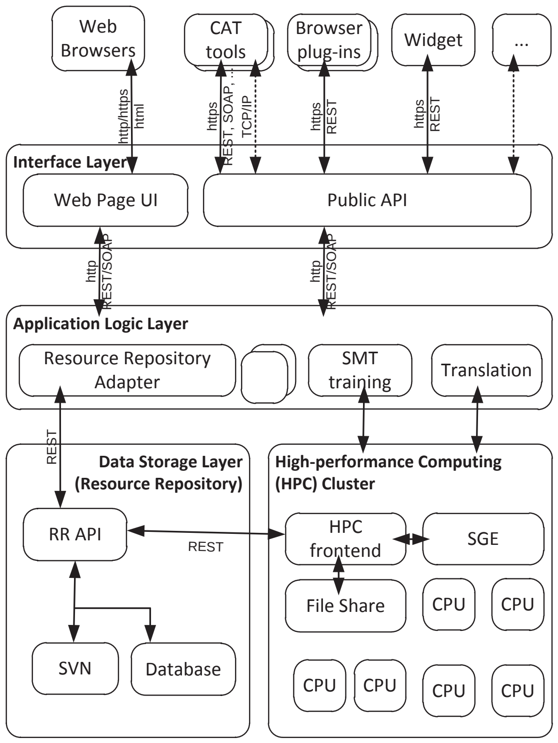


Figure 1. The LetsMT! system architecture

The LetsMT! system has a multitier architecture. It has (i) an interface layer implementing the user interface and APIs with external systems; (ii) an application logic layer for the system logic and (iii) a data storage layer consisting of file and database storage. The LetsMT! system is performing various time and resource consuming tasks; these tasks are defined by the application logic and the data storage and are sent to a High Performance Computing (HPC) Cluster for execution.

The Interface layer provides interfaces between the LetsMT! system and external users. The system has both human and machine users. Human users can access the system through web browsers by using the LetsMT! web page interface. External systems such as CAT tools and browser plug-ins can access the LetsMT! system through a public API. The public API is available through both REST/JSON and SOAP protocol web services. Some CAT tools or other external systems may require different interfaces; they might be introduced if necessary. A HTTPS protocol is used to ensure secure user authentication and secure data transfer.

The application logic layer contains a set of modules responsible for the main functionality or logic of the systems. It receives queries and commands from the interface layer and prepares answers or performs tasks using the data storage and the HPC cluster. This layer contains several modules such as the Resource Repository Manager, the User Manager, the SMT Training Manager etc. The interface layer accesses the application logic layer through both REST/JSON and SOAP protocol web services. The same protocols are used for communication between modules in the application logic layer.

As a data sharing and MT platform the LetsMT! system is able to store large amounts of SMT training data (parallel and monolingual corpora) as well as trained models of SMT systems. The data is stored in one central Resource Repository (RR). The RR is also used to store various tools necessary for data processing and SMT training. As training data may change (for example, grow), the resource repository is based on a version-controlled file system (currently we use SVN as the backend system). A key-value store is used to keep metadata and statistics about training data and trained SMT systems. Modules from the application logic layer and HPC cluster access RR through a REST-based web service interface. Although LetsMT! platform is in beta version now, it is already populated with initial SMT training data. Currently it contains almost 500 million parallel sentences in more than 90 languages.

A High Performance Computing Cluster is used to execute many different data processing tasks, training and running SMT systems. Modules from the application logic and data storage layers create jobs and send them to HPC cluster to execute. HPC cluster is responsible for accepting, scheduling, dispatching, and managing the remote and distributed execution of large numbers of standalone, parallel or interactive jobs. It also manages and schedules the allocation of distributed resources such as processors, memory and disk space. The LetsMT! HPC cluster is based on the Oracle Grid Engine (SGE). The HPC cluster accesses data stored in the data storage layer using the RR API.

The hardware infrastructure of the LetsMT! platform is heterogeneous. The majority of services run on Linux platforms (Giza++, Moses, Resource Repository, data processing tools). The Web server and application logic services run on a Microsoft Windows platform.

The system hardware architecture is designed to be highly sizable. The LetsMT! platform contains several machines with both continuous and on-demand availability:

- Continuous availability – the core frontend and backend services that guarantee LetsMT! webpage and external API availability;
- On-demand availability – training, translation and data import services (HPC cluster nodes); Additional frontend and backend server instances to increase availability.

To ensure scalability of the system whole LetsMT! system including HPC cluster is hosted in Amazon Web Services infrastructure which provides easy access to on demand computing resources.

4 Application of the Moses SMT toolkit

A significant breakthrough in SMT was achieved by the EuroMatrix project. The project objectives included the creation of translation systems for all pairs of EU languages and the development of open source MT technology including research tools, software and data collections. Its result is the improved open source SMT toolkit Moses developed by the University of Edinburgh. The Moses SMT toolkit is a complete statistical translation system distributed under the Lesser General Public License (LGPL). Moses includes all the components needed to pre-process data and to train language and translation models (Koehn et al. 2007). Moses is widely used in the research community and has also reached the commercial sector. While the use of the software is not closely monitored (there is no need to sign a license agreement), Moses is known to be in commercial use by companies such as Systran (Dugast et al. 2009), Asia Online[2], Autodesk (Plitt and Masselot, 2010), Matrixware[3], Adobe, Pangeanic, Logrus (Joscelyne 2010). The LetsMT! project coordinator Tilde bases its free online Latvian MT system on the Moses platform.

LetsMT! uses Moses as a language independent SMT solution and integrates it as a cloud-based service into the LetsMT! online platform. One of

[2] Asia Online. Wikipedia. 2011-08-22. http://en.wikipedia.org/wiki/Asia_Online. (Archived by WebCite® at http://www.webcitation.org/617phHvgD)

[3] Machine Translation at Matrixware. 2011-08-22. http://ir-facility.net/downloads/mxw_factsheet_smt_200910.pdf. (Archived by WebCite® at http://www.webcitation.org/617gTPMb4)

the important achievements of the LetsMT! project will be the adaptation of the Moses toolkit to fit into the rapid training, updating, and interactive access environment of the LetsMT! platform. The SMT training pipeline implemented in Moses currently involves a number of steps that each require a separate program to run. In the framework of LetsMT! this process will be streamlined and made automatically configurable given a set of user-specified variables (training corpora, language model data, dictionaries, tuning sets).

Additional important improvements of Moses that are being implemented by the University of Edinburgh as part of LetsMT!, are the incremental training of MT models, randomised language models (Levenberg et al. 2009), and separate language and translation model servers. We expect some users to add relatively small amounts of additional training data in frequent intervals. The incremental training will benefit from the addition of these data without re-running the entire training pipeline from scratch.

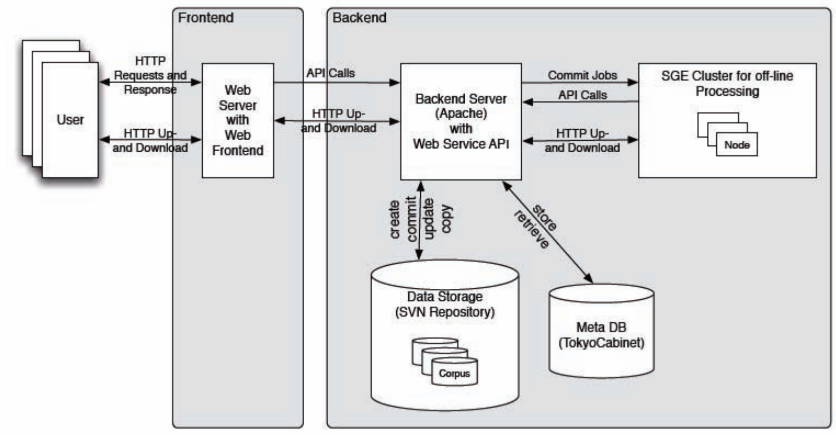
5 LetsMT! Resource repository

Figure 2 illustrates the general architecture of the resource repository and its integration into the LetsMT! platform. The LetsMT! resource repository has a web API that is implemented as a REST service with HTTP requests. The Web API gives access to the LetsMT! resource repository which consists mainly of a revision control system (Subversion), a database (TokyoCabinet) and a batch-queuing system (SGE, Oracle Grid Engine). The purpose of the Web API is to enable the interaction with the repository system for uploading and downloading data, requesting and searching information and triggering batch processes. The LetsMT! resource repository system is implemented in Perl and uses the Apache server and mod_perl to handle the requests and responses to and from the client system.

All data sets of the LetsMT! platform are stored in a revision control system. In the current implementation, we use Subversion (SVN). However, the software is modular and another version control system may replace SVN or even work side-by-side with other storage backends.

Revision control systems are designed for dynamic repositories of textual data in multi-user environments. They typically store all repository modifications and provide tools for tracking the file history for any item in the repository. Furthermore, they naturally support data sharing and possibilities to revert to specific versions. Modifications are stored efficiently by keeping track of changes only. All of this makes them well suited for our needs in which growing resources may be accessed by multiple users.

An important design goal for developing the repository software was to allow arbitrary metadata in terms of key-value pairs stored together with resources in the repository. The focus was set on flexibility in a way that new fields and data sets in various formats can easily be added to the database during development. It has to be possible to store appropriate metadata to any resource at any location in the repository. Another important feature is that this database should still be powerful enough to allow complex search queries over the entire repository which reflects a hierarchical file structure. At the same time, the system has to respect permissions set to individual resources in order to avoid that restricted material can be found. Standard relational database management systems do not support this degree of flexibility as they rely on pre-defined relations (tables) with fixed data types and operations over them. A recent trend is, therefore, to move from relational database with SQL-like queries to schema-less key-value stores that do not


Frontend
Backend
User
HTTP Requests and Response
HTTP Up- and Download
Web Server with Web Frontend
API Calls
HTTP Up- and Download
Backend Server (Apache) with Web Service API
Commit Jobs
API Calls
HTTP Up- and Download
SGE Cluster for off-line Processing
Node
create commit update copy
store retrieve
Data Storage (SVN Repository)
Corpus
Meta DB (TokyoCabinet)
Figure 2. Resource repository overview

require a fixed data model.

A key-value store basically stores arbitrary data (values) by use of a single key. This conceptually simple strategy allows a lot of flexibility in terms of data storage without pre-defined schemas and data models. In relation to the resource repository we like to store arbitrary key-value pairs to any resource in the repository. Various kinds of information shall be stored in this way ranging from descriptive data (textual domain, ownership, language, size etc.) to status information (import/conversion status, etc.), and internal information used by the LetsMT! frontend or repository backend. Furthermore, we also need the support of repeated keys, or better, keys that may contain several values.

Important here is that the database is not restricted to a list of pre-defined keys. In our system, arbitrary keys can be added containing arbitrary values. Furthermore, values can also be interpreted as unordered lists, for example, in the case of language. Using these data sets we are able to ask complex queries such as:

- Give me all public parallel data with English as either source or target language,
- Give me all monolingual data sets from the news domain that are larger than 500 sentences.

Our key-value store is able to process such queries and to return matching resources and their associated metadata entries. Furthermore, we store permission information together with all data records to filter the data according to access restrictions. The backend system we use is based on TokyoCabinet (https://fallabs/tokyocabinet) a freely available software package that implements an efficient database management system with all the flexibility required by our platform.

Another important feature of the Resource Repository software is the support of data import, validation and conversion. Users may upload their data sources in a variety of formats that will automatically be processed by our validation and conversion tools. The software also includes a sentence alignment module that makes it possible to create new parallel resources for SMT training from scratch.

In the current implementation we support the following data formats with dedicated import handlers: aligned parallel data in TMX, XLIFF and Moses formats, monolingual text documents in PDF, Text and DOC formats, compressed data and archives in gzip, zip and tar formats. Support for additional formats may be added in future releases.

6 Conclusions

Current development of SMT tools and techniques has reached the level where they can be implemented in practical applications addressing the needs of large user groups in a variety of application scenarios. The work in progress that is described in this paper promises important advances in the application of SMT by integrating available tools and technologies into an easy-to-use cloud-based platform (https://www.letsmt.eu) for data sharing and generation of customized MT.

The successful implementation of the project will enable wider use and greater impact of available open-source SMT technologies, facilitate diversification of free MT by tailoring it to specific domains and user requirements.

References

L. Dugast, J. Senellart, P. Koehn. 2009. *Selective addition of corpus-extracted phrasal lexical rules to a rule-based machine translation system*, in Proceedings of MT Summit XII

A. Joscelyne. 2010. *TDA Members doing business with Moses*. TAUS DA blog on October 7, 2010. URL: http://www.tausdata.org/blog/2010/10/doing-business-with-moses-open-source-translation/. (Archived by WebCite® at http://www.webcitation.org/617g6iKGN)

P. Koehn, M. Federico, B. Cowan, R. Zens, C. Duer, O. Bojar, A. Constantin, E. Herbst. 2007. *Moses: Open Source Toolkit for Statistical Machine Translation*, in Proceedings of the ACL 2007 Demo and Poster Sessions, pages 177-180, Prague.

A. Levenberg, M. Osborne. 2009. *Stream-based Randomised Language Models for SMT*, in Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing.

F.J. Och, H. Ney. 2003. *A Systematic Comparison of Various Statistical Alignment Models*. Computational Linguistics, (29)1: 19-51.

M. Plitt, F. Masselot. 2010. *A Productivity Test of Statistical Machine Translation Post-Editing in a Typical Localisation Context*. The Prague Bulletin of Mathematical Linguistics, 93(January 2010): 7–16