OPUS-CAT: Desktop NMT with CAT integration and local fine-tuning

Tommi Nieminen
University of Helsinki, Yliopistonkatu 3, 00014 University of Helsinki, Finland
tommi.nieminen@helsinki.fi

Abstract

OPUS-CAT is a collection of software which enables translators to use neural machine translation in computer-assisted translation tools without exposing themselves to security and confidentiality risks inherent in online machine translation. OPUS-CAT uses the public OPUS-MT machine translation models, which are available for over a thousand language pairs. The generic OPUS-MT models can be fine-tuned with OPUS-CAT on the desktop using data for a specific client or domain.

1 Introduction

Neural machine translation (NMT) has brought about a dramatic increase in the quality of machine translation in the past five years. The results of the latest European Language Industry Survey (FIT Europe et al., 2020) confirm that NMT is now routinely used in professional translation work. NMT systems used in translation work are developed by specialized machine translation vendors, translation agencies, and organizations that have their own translation departments. Translators use NMT either at the request of a client, in which case the client provides the NMT, or independently, in which case they usually rely on web-based services offered by large tech companies (such as Google or Microsoft) or specialized machine translation vendors. These web-based services are mainly used through machine translation plugins or integrations that are available for all major computer-assisted translation (CAT) tools, such as SDL Trados and memoQ.

Even though MT has been extensively used in the translation industry for over a decade (Doherty et al., 2013), there is still considerable scope for growth: according to FIT Europe et al. (2020), 78 percent of language service companies plan to increase or start MT use, and most independent translation professionals use MT only occasionally. One of the factors slowing down the adoption of MT are risks related to confidentiality and security. There are well-known risks involved with using web services, which also concern the web-based NMT services available to translators and organizations: data sent to the service may be intercepted en route, or it may be misused or handled carelessly by the service provider. These security and confidentiality risks (even if they are unlikely to actualize) hinder MT use by independent translation professionals, since their clients often specifically forbid or restrict the use of web-based MT (European Commission, 2019). Even if using web-based MT is not expressly forbidden, translators may consider it unethical or they may fear it might expose them to unexpected legal liabilities (Kamocki et al., 2016).

Producing MT directly on the translator's computer without any communication with external services eliminates the confidentiality and security risks associated with web-based MT. This requires an optimized NMT framework which is capable of running on Windows computers (as most CAT tools are only available for Windows), and pre-trained NMT models for all required language pairs. The Marian NMT framework (Junczys-Dowmunt et al., 2018) fulfills the first requirement, as it is highly optimized and supports Windows builds. Pre-trained NMT models are available from the OPUS-MT project (Tiedemann and Thottingal, 2020), which trains and publishes Marian-compatible NMT models with the data collected in the OPUS corpus (Tiedemann, 2012). OPUS-CAT is a software collection which contains a local MT engine for Windows computers built around the Marian framework and OPUS-MT models, and a selection of plugins for CAT tools. OPUS-CAT is aimed at professional translators, which is why it also supports the fine-tuning of the base OPUS-MT models with

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations, pages 288–294
April 19 - 23, 2021.

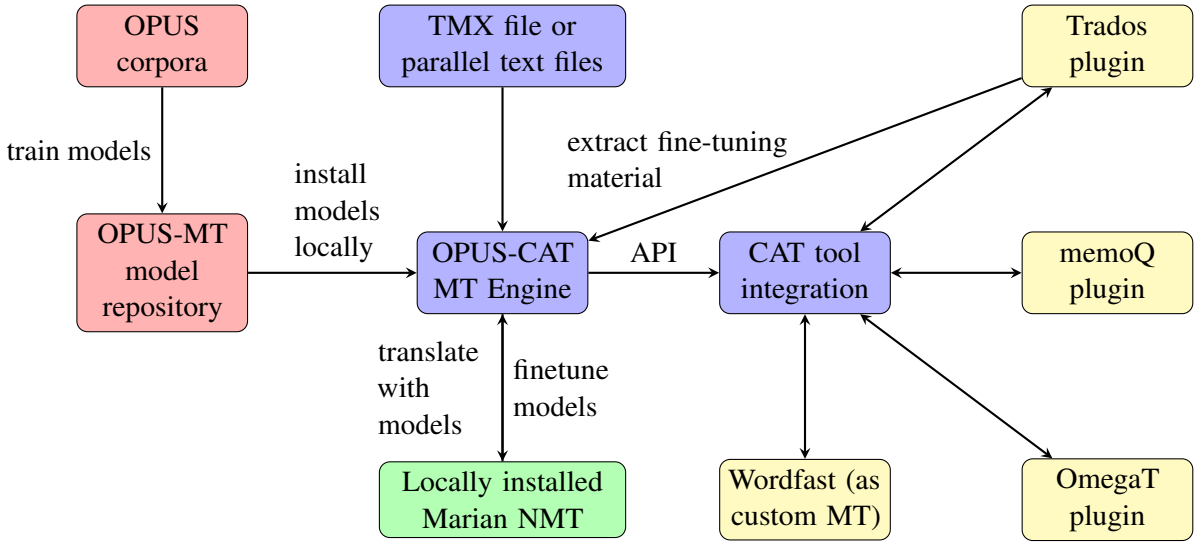

Figure 1: Diagram of the software and models used in OPUS-CAT.

project-specific data.

2 OPUS-CAT MT Engine

The main component of OPUS-CAT is the OPUS-CAT MT Engine, a locally installed Windows application with a graphical user interface. OPUS-CAT MT Engine can be used to download NMT models from the OPUS-MT model repository, which contains models for over a thousand language pairs.

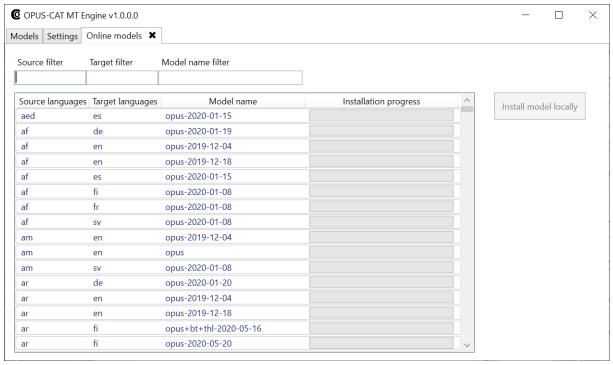

Figure 2: Install OPUS-MT models locally (1,000+ language pairs available)

Once a model has been downloaded, OPUS-CAT MT Engine can use it to generate translations by invoking a Marian executable included in the installation. Before the text is sent to the Marian executable, OPUS-CAT MT Engine automatically pre-processes the text using the same method that was originally used for pre-processing the training corpus of the model. Pre-processing is model-specific, as the older OPUS-MT models use Subword NMT (Sennrich et al., 2016) for segmenting the text while newer models use SentencePiece (Kudo and Richardson, 2018). Both Subword NMT and SentencePiece are coded in Python, but they are distributed as standalone Windows executables with OPUS-CAT MT Engine, as requiring the users to install Python in Windows would complicate the setup process.

OPUS-CAT MT Engine user interface provides a simple functionality for translating text, but the translations are mainly intended to be generated via an API that the OPUS-CAT MT Engine exposes. This API can be used via two protocols: net.tcp and HTTP. net.tcp is used with plugins for the SDL Trados and memoQ CAT tools, while HTTP is used for other plugins and integration. The motivation for using net.tcp is that exposing a net.tcp service on the local Windows computer does not require administrator privileges, which makes setting up the OPUS-CAT MT Engine much easier for non-technical users. However, Trados and memoQ are the only CAT tools with sufficiently sophisticated plugin development kits to allow for net.scp connections, so the API can also be used via HTTP with some extra configuration steps, so that it can be used from other tools. The API has three main functionalities:

- **Translate**: Generates a translation for a source sentence (or retrieves it from a cache) and returns it as a reply to the request.
- **PreorderBatch**: Adds a batch of source sentences to the translation queue and immediately returns a confirmation without waiting for the translations to be generated.

- **Customize**: Initiates model customization using the fine-tuning material included in the request.

The OPUS-CAT MT Engine stores the local NMT models in the user's application data folder in order to avoid file permission issues. Local application data folder is used, as saving the models in the roaming application data folder could lead to unwanted copying of the models, if same user profile is used on multiple computers. The user interface of the OPUS-CAT MT Engine contains functionalities for managing models installed on the computer, such as deletion of models, packaging of models for migration to other systems, and tagging the models with descriptive tags. The tags can be used to select specific models in CAT tool plugins, e.g. a model fine-tuned for a specific customer can be tagged with the name of the customer.

3 CAT tool plugins and integration

OPUS-CAT contains plugins for three CAT tools: SDL Trados, memoQ and OmegaT. OPUS-CAT can also be used with the Wordfast CAT tool, which supports fetching translations from services via APIs. As SDL Trados is the established market leader among CAT tools, and it has the most extensive plugin development support, the OPUS-CAT plugin for SDL Trados is more feature-rich than the other plugins. The other plugins simply support fetching translations through the Translate API method. SDL Trados plugin also contains an option to initiate the fine-tuning of a model based on the bilingual material included in a translation project.

One difficult aspect of integrating MT services with CAT tools is latency. Delays in presenting the translation to the user affect the user experience adversely and may even lower productivity significantly. For most MT services the delay is due to web latency, but for OPUS-CAT the generation of translations itself may be so slow that it causes a visible delay, since OPUS-CAT uses a CPU for translation instead of a much faster GPU. In any CAT tool, this delay can be eliminated by pre-translating the translation project with the MT service prior to starting the translation.

In the OPUS-CAT plugin for SDL Trados there is also a feature which can be used to initiate the translation of segments ahead of time. Whenever the translator moves to a new segment, the plugin will send the segments following the selected segment (the number of segments can be configured in the plugin settings) to the OPUS-CAT for translation. This means that when the translator moves to the next segment, it has already been translated and can simply be retrieved from the OPUS-CAT translation cache.

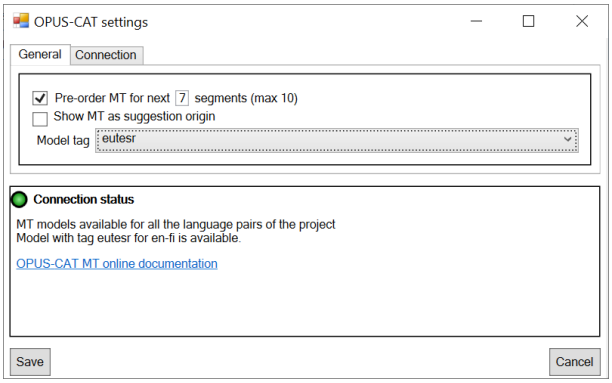

Figure 3: Trados plugin settings. Note the preordering function and model tag.

4 Local fine-tuning of models

OPUS-CAT is intended for professional translators, and the utility of generic NMT models in professional translation is uncertain (Sánchez-Gijón et al., 2019), while performance improvements resulting from the use domain-adapted NMT models have been observed multiple times (Läubli et al., 2019; Macken et al., 2020). Because of this, OPUS-CAT MT Engine includes a functionality for fine-tuning models with small amounts of bilingual data. The method of fine-tuning is simple: the generic OPUS-MT model for a language pair is duplicated, and Marian training of the model is resumed with the model using the domain-specific data as the training set. This particular method of fine-tuning was first described in Luong and Manning (2015), but adaptation of statistical and neural MT models with domain-specific data has been common for over a decade (Koehn and Schroeder, 2007). The fine-tuning is performed using the same Marian executable included with OPUS-CAT MT Engine installation, which is also used for generating translations.

OPUS-CAT MT Engine is a Windows program intended to run on mid-tier desktop and laptop computers that translators commonly use, so the fine-tuning process cannot be computationally intensive. The fine-tuning must rely on CPUs, since GPUs suitable for neural network training are not available on the translators' computers. This places severe restrictions on the size of the fine-tuning set

and the duration of the training. Furthermore, the fine-tuning functionality is intended to be used by translators without specialist knowledge about machine translation, so the users cannot be expected to be able to adjust the fine-tuning settings. That is why the fine-tuning functionality has to have a conservative set of default settings that work in almost all environments and circumstances.

In a typical translation job, deadlines usually allow for considerably more time for delivery of the translation than the actual translation work requires. This is due to the fact that translators normally have multiple jobs underway or lined up at any given time, and extended deadlines are required so that translators can organize and prioritize their work. This means that there is generally at least a couple of hours of time available for running the fine-tuning process before the actual translation work has to begin. On the basis of this estimate, the fine-tuning process should generally take at most two hours.

Another consideration in fine-tuning is that the process takes place on a computer that may be used for other tasks during the fine-tuning. This means that the fine-tuning process cannot take advantage of all the resources available on the computer, as doing so would cause performance issues for the user. In order to keep the fine-tuning as non-intrusive as possible, OPUS-CAT MT Engine uses only a single thread and a workspace of 2048 MB for fine-tuning.

Because of the limited amount of processing power available and the target duration of at most two hours, the fine-tuning is stopped after a single epoch by default. The actual duration will vary according to the sentence count and the sentence length distribution of the fine-tuning set. The duration will be approximately two hours when the fine-tuning set contains the default maximum amount of sentences, which is 10,000.

It would be possible to allow the fine-tuning to last for multiple epochs and to adjust the amount of epochs based on the sentence count, but informal testing during development indicated that a single epoch of fine-tuning tends to have a noticeable effect on the MT output even with small fine-tuning sets. Also, some output corruption indicating over-fitting was detected when fine-tuning was continued over many epochs. Because of these indications of over-fitting, the learning rate was also lowered to 0.00002 from the default 0.0001. Advanced users can change these default settings in the settings tab of the OPUS-CAT MT Engine.

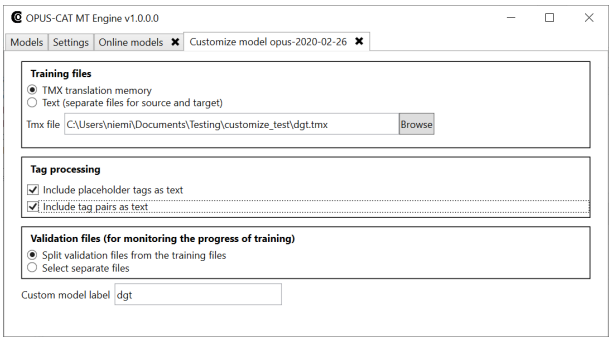

Figure 4: Initiating fine-tuning from OPUS-CAT MT Engine.

Currently fine-tuning can be initiated directly from the OPUS-CAT MT Engine or from the SDL Trados plugin. When initiated from the OPUS-CAT MT Engine, a .tmx file or a pair of source and target files can be used as fine-tuning material. Fine-tuning from the SDL Trados plugin allows for much more sophisticated selection of fine-tuning material. The fine-tuning functionality in the SDL Trados plugin is implemented as a batch task, which is performed for a given translation project. Translation projects usually contain segments which have already been translated (full matches). By default, the fine-tuning task extracts these segments as fine-tuning material. These are assumed to be the most relevant material, since they pertain directly to the translation project.

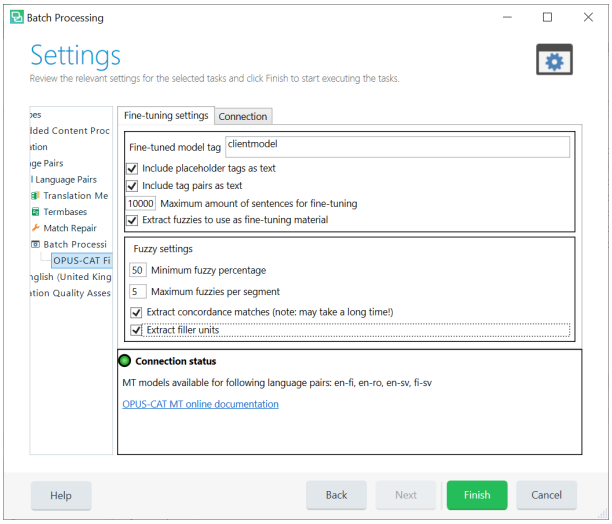

Figure 5: Initiating fine-tuning from OPUS-CAT plugin for SDL Trados.)

If the translation project does not contain enough full matches, it is possible to extract translation

units from the translation memories attached to the translation project. The fine-tuning task uses the fuzzy matching functionality of SDL Trados to extract partially matching translation units according to the specified minimum fuzzy percentage. The task can also extract fine-tuning material by performing a concordance search for words or sequences of words in the source segment. Finally, if the other methods have not managed to extract a sufficient amount of fine-tuning material, the task can simply bulk up the fine-tuning set by extracting segments from the translation memories, starting with the newest (assumed to be the most relevant).

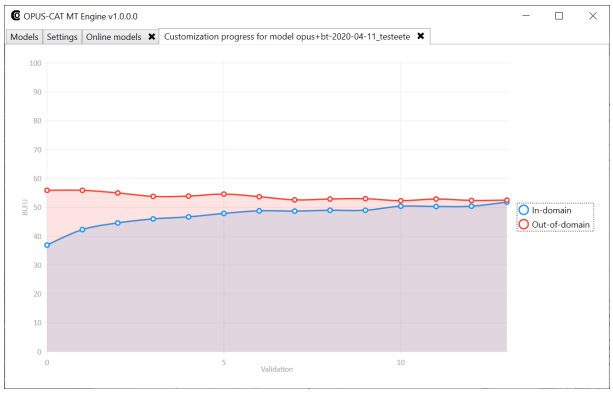

Figure 6: Monitoring fine-tuning progress.)

Progress of the fine-tuning can be monitored in the OPUS-CAT MT Engine. When a fine-tuning job is initiated, part of the fine-tuning set is separated for use as an in-domain validation set. For most language pairs, OPUS-CAT MT Engine also contains out-of-domain validation sets, which have been extracted from the Tatoeba corpus (Tiedemann, 2020). The in-domain and out-of-domain validation sets are combined and evaluated periodically during fine-tuning with SacreBLEU (Post, 2018), which is also included in OPUS-CAT as a standalone Windows executable. The evaluation results for each set are plotted as a graph in the OPUS-CAT MT Engine. OPUS-CAT MT Engine also displays an estimate of the remaining duration of the fine-tuning. These visual indications of progress are important, as the users of the fine-tuning functionality are translators without specialist technical skills.

The fine-tuning functionality also has an option to include tags found in the fine-tuning set as generic tag markers. The fine-tuned model will then learn to generate tag markers in the translations, and these can be used to transfer tags from source to target in CAT tools (currently only the SDL Trados plugin supports tag conversion). Placeholder tags and tag pairs are converted separately. This approach to tag handling is similar to the one found in (Hanneman and Dinu, 2020), but simpler. The main difference is that the same textual tag marker is used for every tag, so the tag handling assumes that the tag order is identical in both source and target.

5 Related work

Desktop MT systems have been available at least since the 1990s (Richards, 1994), when desktop computers became powerful enough to run rule-based MT systems. In the SMT era, the higher computational requirements made desktop MT difficult, but there were still some examples of desktop SMT, such as (Slate Rocks!, 2021). Unlike OPUS-CAT, these earlier desktop MT programs were commercial products. As for NMT, the currently active Bergamot project (Bergamot, 2021) aims to make client-side MT available in web browsers, and also uses Marian as its NMT framework. However, Bergamot is aimed at the common public, while OPUS-CAT is intended for professional translators. To our knowledge, there is no other software that is free to use and offers a local NMT fine-tuning functionality (commercial MT providers do provide local MT engine installations, which may support local fine-tuning).

6 Current status and future work

OPUS-CAT is based on software developed originally for the Fiskmö project (Tiedemann et al., 2020), and it is currently being developed as part of the European Language Grid programme. The previous version of the software has been used by several organizations in Finland for professional translation. Based on the feedback from users, the most important features that translators would like to see are real-time adaptation of NMT models with new translations, and the enforcement of correct terminology and document-level consistency. These will be the main priorities in the development of OPUS-CAT. We will also be collecting user experiences on the local fine-tuning capability, and will develop the feature and its documentation according to that feedback.

7 Conclusion

OPUS-CAT is collection of software that makes it possible to use NMT locally on desktop computers

without risks posed by web-based services. It uses models from the OPUS-MT project, which offers NMT models for over a thousand language pairs. OPUS-CAT is based on the efficient and optimized Marian NMT framework, which is fast enough to work usefully even with mid-tier computers. The local fine-tuning functionality makes it possible to adapt models to specific domains and clients, which is vital when using MT for professional translation. OPUS-CAT also contains plugins for several major CAT tools, and exposes an API which can be used in integrations with other tools. The OPUS-CAT plugin SDL Trados is especially well suited for integration into translation workflows due to its sophisticated fine-tuning functionality, which is implemented as a workflow task. OPUS-CAT is licensed under the MIT License, and the source code and software releases are available at `https://github.com/Helsinki-NLP/OPUS-CAT`.

References

Bergamot. 2021. Bergamot.

Stephen Doherty, Federico Gaspari, Declan Groves, Josef Genabith, Lucia Specia, Arle Lommel, Aljoscha Burchardt, and Hans Uszkoreit. 2013. Mapping the industry i: Findings on translation technologies and quality assessment. *Globalization and Localization Association*.

European Commission. 2019. Tender specifications: Translation of european union documents.

FIT Europe, EUATC, ELIA, GALA, and LINDWeb. 2020. European language industry survey 2020.

Greg Hanneman and Georgiana Dinu. 2020. How should markup tags be translated? In *Proceedings of the Fifth Conference on Machine Translation*, pages 1160–1173, Online. Association for Computational Linguistics.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. 2018. Marian: Fast neural machine translation in C++. In *Proceedings of ACL 2018, System Demonstrations*, Melbourne, Australia.

Pawel Kamocki, Jim O'Regan, and Marc Stauch. 2016. *All Your Data Are Belong to us . European Perspectives on Privacy Issues in 'Free' Online Machine Translation Services*.

Philipp Koehn and Josh Schroeder. 2007. Experiments in domain adaptation for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 224–227, Prague, Czech Republic. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. pages 66–71.

Samuel Läubli, Chantal Amrhein, Patrick Düggelin, Beatriz Gonzalez, Alena Zwahlen, and M. Volk. 2019. Post-editing productivity with neural machine translation: An empirical assessment of speed and quality in the banking and finance domain. In *MT-Summit*.

Minh-Thang Luong and Christopher D. Manning. 2015. Stanford neural machine translation systems for spoken language domains.

Lieve Macken, Daniel Prou, and Arda Tezcan. 2020. Quantifying the effect of machine translation in a high-quality human translation production process. *Informatics*, 7.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

John Richards. 1994. LogoVista E to J. In *Proceedings of the First Conference of the Association for Machine Translation in the Americas*, Columbia, Maryland, USA.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Slate Rocks! 2021. Slate desktop.

Pilar Sánchez-Gijón, Joss Moorkens, and Andy Way. 2019. Post-editing neural machine translation versus translation memory segments. *Machine Translation*, 33:1–29.

Jörg Tiedemann. 2020. The Tatoeba Translation Challenge – Realistic data sets for low resource and multilingual MT. In *Proceedings of the Fifth Conference on Machine Translation (Volume 1: Research Papers)*. Association for Computational Linguistics.

Jörg Tiedemann, Tommi Nieminen, Mikko Aulamo, Jenna Kanerva, Akseli Leino, Filip Ginter, and Niko Papula. 2020. The FISKMÖ project: Resources and tools for Finnish-Swedish machine translation and cross-linguistic research. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 3808–3815, Marseille, France. European Language Resources Association.

Jörg Tiedemann and Santhosh Thottingal. 2020. OPUS-MT — Building open translation services for the World. In *Proceedings of the 22nd Annual Conferenec of the European Association for Machine Translation (EAMT)*, Lisbon, Portugal.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in opus. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).