PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 106 OCTOBER 2016 147–158

Qualitative: Python Tool for MT Quality Estimation Supporting Server Mode and Hybrid MT

Eleftherios Avramidis

German Research Center for Artificial Intelligence (DFKI) Berlin, Germany

Abstract

We are presenting the development contributions of the last two years to our Python open-source Quality Estimation tool, a tool that can function in both experiment-mode and online web-service mode. The latest version provides a new MT interface, which communicates with SMT and rule-based translation engines and supports on-the-fly sentence selection. Additionally, we present an improved Machine Learning interface allowing more efficient communication with several state-of-the-art toolkits. Additions also include a more informative training process, a Python re-implementation of QuEst baseline features, a new LM toolkit integration, an additional PCFG parser and alignments of syntactic nodes.

1. Introduction

After almost a decade of research, evaluating Machine Translation (MT) remains an active topic. Through the years, a multitude of methods have been researched, in a continuous effort to assess whether the MT output adheres to the users expectations. For a significant amount of time, ranking has been the dominant approach for MT equality, since it seems a relatively robust way to circumvent the subjectivity of perceiving quality (Callison-Burch et al., 2007, 2008; Bojar et al., 2015, etc.).

Many automatic metrics have been developed in order to measure MT quality by comparing it to the reference translations (e.g. Papineni et al., 2002), facing the limitation that the reference represents usually only one of the possible translations. As a more recent development, Quality Estimation (QE) has shifted the focus from the reference translations towards the translations themselves, by identifying qualitative features that can be indications of a good translation.

 Corresponding author: eleftherios.avramidis@dfki.de
Cite as: Eleftherios Avramidis. Qualitative: Python Tool for MT Quality Estimation Supporting Server Mode and Hybrid MT. The Prague Bulletin of Mathematical Linguistics No. 106, 2016, pp. 147–158.
doi: 10.1515/pralin-2016-0014.

The work presented here is a programming effort to ease research in both aspects sketched above. We present the latest version of Qualitative (Avramidis et al., 2014), a QE tool that processes translations as a ranking, in an attempt to learn better the human preferences. At the same time, extensive engineering takes place to devise new features by enabling various natural language processing (NLP) methods.

The version of the toolkit presented here is a result of more than two years of development and offers a data processing unit, a powerful feature generation engine with feature selection, a machine learning interface and a collection of evaluation metrics. Additionally, it can perform hybrid machine translation by communicating with several MT engines and combining their output on a sentence level. The development takes place in GitHub [1] and further contributions are welcome.

2. Related work

Few pieces of software on QE have been released with an open source. QuEst (Specia et al., 2013), previously also known as Harvest, is the most established one, as it has been used as a baseline for the yearly WMT Shared Tasks on QE (e.g. Callison-Burch et al., 2012). The main difference with our approach is that it uses two different pieces of software for feature generation and machine learning, where the former is written in Java and the latter in Python. Additionally, many parts of it operate only in batch mode. For these two reasons, in contrast to our software, operating in a real-usage scenario (e.g. server mode) with sentence-level requests is non-trivial. Its latest version, QuEst++ (Specia et al., 2015), additionally supports word-level and document-level QE.

Some most recent software focuses on an another level of granularity, namely word-level QE. WceLig (Servan et al., 2015) is a tool which introduces support for various target languages, handles glass-box, lexical, syntactic and semantic features for estimating confidence at word-level. Marmot (Logacheva et al., 2016), focuses on word-level and phrase-level QE and is written in Python. It offers a modular architecture, users can easily add or implement new parsers, data representations and features that fit their particular use cases, whereas it can be easily plugged into a standard experiment workflow.

In contrast to most of the above software, the approach of the software presented here focuses on a double-usage scenario for both scientific experimentation and real-usage. Feature generators and machine learning support both batch mode and sentence-level mode, whereas the functionality can be easily plugged into web-services and other software that requires QE functionality. Furthermore, it offers a dynamic pipeline architecture, including wrappers for NLP tools written in several programming languages.

[1] The source code, along with basic documentation for installation, execution and further development can be found at `https://github.com/lefterav/qualitative`

3. Design

The software has been developed based on a multilateral design that serves the operational requirements sketched above. This section includes the main architecture of the pipeline and the interoperability features with embedded software.

3.1. Architecture

The software consists of:

- a **data processing unit** able to read XML and text-based input,
- a **pre-processing stage** that performs the necessary string normalisation process for the languages at hand,
- a **machine translation** module, which communicates with external MT systems and handles sentence-level system combination,
- a **feature generation engine** that produces hundreds of dynamic black-box and glass-box features, by harvesting the output of embedded open-source NLP tools,
- a **machine learning** interface that embeds widely-used ML libraries, including data conversion to their internal structures. Additionally there are pairwise wrappers that allow the usage of binary classifiers for ranking and
- an **evaluation package** that includes several metrics for measuring ranking and translation performance.

3.2. Interoperability

A detailed diagram of the architecture can be seen in Figure 1. Additionally to the core architecture which is seen in the horizontal axis, the system integrates external components developed in various programming languages. These 25 components are integrated using 9 different approaches, including native Python libraries, sockets to the Java Virtual Machine (JVM), wrappers, system pipes and remote service APIs (e.g. JSON, REST).

The majority of these tools are seemlessly integrated and available as callable Python objects throughout the entire pipeline. For instance, Truecasing (Wang, Wei and Knight, Kevin and Marcu, 2006) is done with the original Moses scripts via Perl pipes, features from PCFG parsing are collected through a JVM socket from Berkeley Parser (Petrov et al., 2006), whereas Machine Translation is fetched from Moses (Koehn et al., 2006) via XML-RPC. More details on the interoperability interfaces can be found in Avramidis (2016).

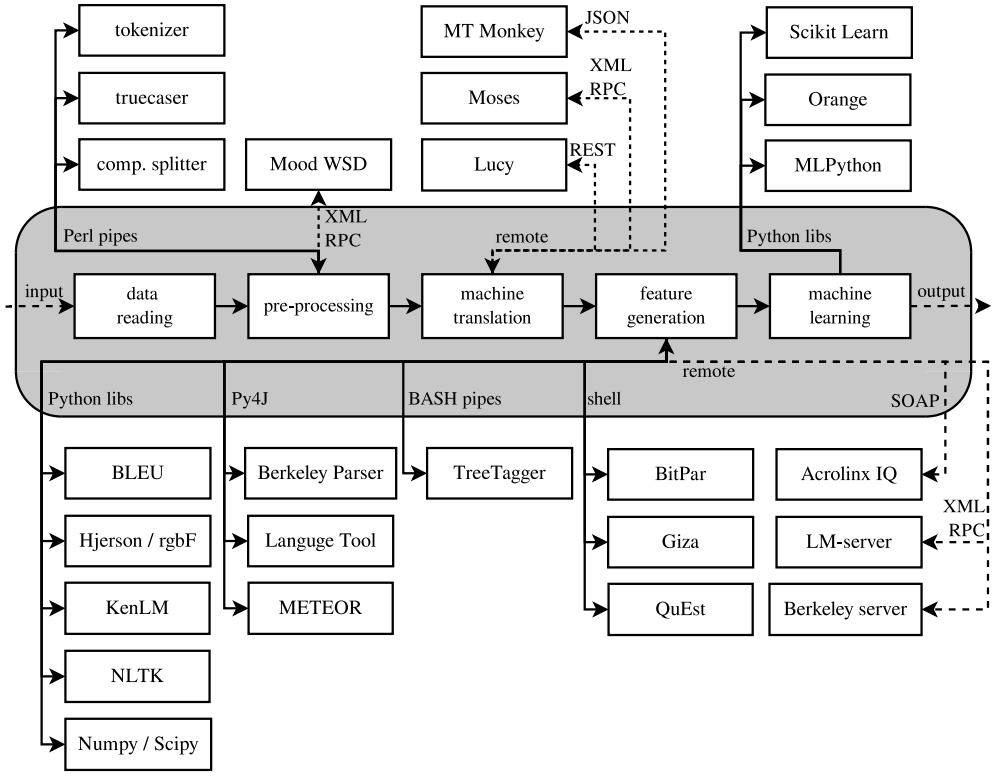

Figure 1. Full diagram of the components that have been integrated into the application. Source: (Avramidis, 2016)

```
<?xml version="1.0" encoding="utf-8"?> <jcml>
  <judgedsentence langsrc="en" id="11" langtgt="de">
    <src>Go to System Preferences</src>
    <tgt system="pilot_0" berkley-loglikelihood="-84.9089431054"
         berkeley-n="19" rank="2">Gehen Sie zu Systemeinstellungen</tgt>
    <tgt system="pilot_2" berkley-loglikelihood="-74.6569913407"
         berkeley-n="5" rank="3">Sprung zu Systemeinstellungen</tgt>
    <ref berkley-loglikelihood="-83.3551531463"
         berkeley-n="18" >Gehen Sie zu Systemeinstellungen</ref>
  </judgedsentence>
</jcml>
```

Figure 2. Sample JCML file, containing a source sentence, the reference and two translations with Berkeley Parser scores and human ranks

4. New functionality

For the generic functionality, including instructions on how to run the tool, the reader is referred to (Avramidis et al., 2014) and for the underlying methodology to Avramidis (2012a). Here, we outline the functionality introduced in the latest version.

4.1. Data Processing

The majority for the read/write processing of the software is done in a special XML format, the JCML format, which stands for *Judged Corpus Markup Language*. It is a simple XML format that has been devised so that it allows dynamic feature lists but at the same time it can be inspected manually. The latest improvements include incremental reading and writing, a feature which solved many memory-related issues, given the big volume of some data sets. There are also several scripts that allow the conversion from and to other common formats. A sample of such a file can be seen in Figure 2.

4.2. Machine translation

One of the basic applications of the automatic ranking is the possibility to combine differen systems on the sentence level. Such a method is often referred to as a case of *hybrid* MT when it combines different types of systems (e.g. statistical and rule-based). This version offers a new package that handles the communication with translation engines by connecting to remote APIs. It currently supports Moses (Koehn et al., 2006), Lucy (Alonso and Thurmair, 2003), as well as MT-Monkey (Tamchyna et

al., 2013) for accessing deployed server installations. The communication with the engines allows fetching translations and glass-box features (translation probability, unknown words etc.), when these are made available by the engine.

Additionally, some techniques of hybridisation are included, such as serial post-editing of a rule-based system with a statistical system (SMT) (as implemented for Avramidis et al., 2015), partial translation of terminology from the rule-based system with SMT, and SMT including an alternative decoding path from WSD disambiguation (Avramidis et al., 2016).

The machine translation interface, apart from being a step of the QE pipeline, it can also act as a standalone application, or as a web-service pluggable via XML-RPC.

4.3. Feature Generation

The modular interface of the feature generation pipeline allows easy plugging of new Feature Generators. These are classes which process the text of the sentences and return numerical values that describe some aspects of quality. The existing functionality, presented in the past, includes usage of language models, PCFG parsing, cross-target BLEU and METEOR (Banerjee and Lavie, 2005), language correction, IBM-1 probabilities, as well as token counts.

The new version offers additionally:

- **word alignment** based on the IBM-1 model (Brown et al., 1993), allowing to derive the count of aligned PCFG tree spans, nodes and leaves between the source and the target sentence. Whereas this generates hundreds of sparse features, the most prominent of them are expected to help isolate systems that fail to translate grammatically important chunks of the source,
- relative and absolute **position** of every PCFG tag within the sentence, with the goal to capture wrong positioning of grammatical chunks in languages where this is important (e.g. German),
- a re-implementation of the **WMT baseline features** (Callison-Burch et al., 2012) in Python, including the average number of translations per source word in the segment as given by IBM-1 model with probabilities thresholded in different ways, and the average number of occurrences of the target word within the target segment,
- integration of **KenLM** (Heafield, 2011) via its Python library, which allows efficient of loading compiled language models, removing the previous requirement for an external LM server,
- a wrapper for the PCFG parser **BitPar** (Schmid, 2004), as an alternative for Berkeley Parser (integration based on van Cranenburgh, 2010; van Cranenburgh et al., 2010),
- a wrapper for the **TreeTagger** (Schmid, 1994), which acquires the necessary POS tags for Hjerson (Popović, 2011)

- a connector to the XML-RPC of MoodWSD (Weissenborn et al., 2015), an external word sense disambiguator

4.4. Machine Learning

A new more transparent and modular internal interface allows for integration of several external machine learning (ML) toolkits. The integration of every ML toolkit should extend an abstract class named `Ranker`. This should implement some basic functions, such as training on a batch of sentences, or producing the ranking given one source sentence and its translations. The implementation of every ML toolkit is also responsible of converting the given sentence data and its features to the data structures understood by the toolkit. Binary classifiers, where available, are wrapped to provide a ranker's functionality.

Currently the following toolkits and learning methods are supported:

- Orange (Demšar et al., 2004) with k-Nearest Neighbours, Logistic Regression with Stepwise Feature Set Selection or L2-regularisation and C45 trees,
- SciKit Learn (Pedregosa et al., 2011) with Support Vector Machines with Grid parameter optimisation over cross-validation, Decision Trees, Gaussian Naïve Bayes, Linear and Quadratic Discriminant Analysis, Bagging, Adaptive Boosting and Gradient Boosting and feature selection methods such as Recursive Feature Elimination with Cross-Validation
- MLpython[2] with listwise ranking methods, such as ListNet (Cao et al., 2007).

4.5. Evaluation

The evaluation phase is the last part of the experiment process, as the trained models are tested against gold-sets and need to be evaluated accordingly. The evaluation phase offers a wide range of ranking metrics, with latest additions the inverse-weighed Kendall's τ and its theoretical p-values and confidence intervals. Finally, the evaluation phase includes automatic metric scores (BLEU, METEOR, TER, WER, Hjerson) for the performance of the system combination and its components against the reference translation.

4.6. Experimental management

Similarly to the previous version, experimenting over the training of new models is organised by using the ExpSuite (Rückstieß and Schmidhuber, 2011). This allows the exhaustive exploration of several experimental settings in parallel. We have extended the functionality to provide out-of-the-box parellelised **cross-validation** for any given dataset. Additionally, the split training and test-sets of the cross-validation are **cached**

[2]MLpython is described at `http://www.dmi.usherb.ca/~larocheh/mlpython/`

in a common directory, so that they can be re-used for different experimental settings which require the same original dataset. The experiments can be resumed from the step they were left, in case of any unexpected interruption.

The experiment pipeline keeps a structured log of every step of the experiment, which may include the results of the evaluation, but also details about the machine learning process (e.g. the beta coefficients of a log-linear model, or weights of a linear model). The trained models are also dumped in external files, so that they can be re-used later. After all iterations and cross-validation folds of the experiment are concluded, a script allows for creating a comma-separated table that compares all experimental settings against a desired set of metrics.

5. Further work

There are often upgrades to the integrated external software that fix issues or provide additional functionality. Although sticking to older tested versions usually suffices, further work may include adaptations for newer versions of this software. In this direction, adjusting the current Python 2.7 code to support Python 3 would be useful.

Whereas the interface for machine learning over ranking has been re-developed as outlined above, most parts of the pipeline have been used for other types of quality estimation, such as quality score prediction for single outputs (Avramidis, 2012b) and error prediction (Avramidis and Popovic, 2014). Small extensions to provide abstract classification and regression interfaces for all ML toolkits would be desirable.

We are also aware that the glass-box feature integration requires extensions to support most MT-engines, although this faces the barrier that not all glass-box features are easily available.

Finally, big-amounts of data, despite the huge potential for machine learning, create bottlenecks in case they must be analyzed or processed selectively. We plan to support more effective data types (e.g. JSON). A possible solution would include the implementation of smart databases and other data-effective techniques.

Acknowledgements

This work has received support by the EC's FP7 (FP7/2007-2013) under grant agreement number 610516: "QTLeap: Quality Translation by Deep Language Engineering Approaches". Early stages have been developed with the support of the projects TaraXÚ and QT-Launchpad. Many thanks to: Slav Petrov for modifying the Berkeley Parser in order to allow modification of parsing parameters; Andreas van Cranenburgh, Hieu Hoang, Philipp Koehn, Nitin Madnani, Laurent Pointal, Maja Popović, Josh Schroeder and David Vilar as parts of their open source code have been included in some of our scripts.

Bibliography

Alonso, Juan A and Gregor Thurmair. The Comprendium Translator system. In *Proceedings of the Ninth Machine Translation Summit*. International Association for Machine Translation (IAMT), 2003.

Avramidis, Eleftherios. Comparative Quality Estimation: Automatic Sentence-Level Ranking of Multiple Machine Translation Outputs. In *Proceedings of 24th International Conference on Computational Linguistics*, pages 115–132, Mumbai, India, dec 2012a. The COLING 2012 Organizing Committee. URL http://www.aclweb.org/anthology/C12-1008.

Avramidis, Eleftherios. Quality estimation for Machine Translation output using linguistic analysis and decoding features. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 84–90, Montréal, Canada, jun 2012b. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W12-3108.

Avramidis, Eleftherios. Interoperability in MT Quality Estimation or wrapping useful stuff in various ways. In *Proceedings of the LREC 2016 Workshop on Translation Evaluation: From Fragmented Tools and Data Sets to an Integrated Ecosystem*, pages 1–6. Language Science Press, 2016. URL http://www.cracking-the-language-barrier.eu/wp-content/uploads/Avramidis.pdf.

Avramidis, Eleftherios and Maja Popovic. Correlating decoding events with errors in Statistical Machine Translation. In Sangal, Rajeev, Jyoti Pawar, and Dipti Misra Sharma, editors, *Proceedings of the 11th International Conference on Natural Language Processing. International Conference on Natural Language Processing (ICON-2014), 11th International Conference on Natural Language Processing, December 18-21, Goa, India*. International Institute of Information Technology, Natural Language Processing Association, India, 2014. URL https://www.dfki.de/lt/publication{_}show.php?id=7605.

Avramidis, Eleftherios, Lukas Poustka, and Sven Schmeier. Qualitative: Open source Python tool for Quality Estimation over multiple Machine Translation outputs. *The Prague Bulletin of Mathematical Linguistics*, 102:5–16, 2014. URL http://ufal.mff.cuni.cz/pbml/102/art-avramidis-poustka-schmeier.pdf.

Avramidis, Eleftherios, Maja Popovic, and Aljoscha Burchardt. DFKI's experimental hybrid MT system for WMT 2015. In *Proceedings of the Tenth Workshop on Statistical Machine Translation. Workshop on Statistical Machine Translation (WMT-2015), 10th, September 17-18, Lisbon, Portugal*, pages 66–73. Association for Computational Linguistics, 2015. URL http://aclweb.org/anthology/W15-3004.

Avramidis, Eleftherios, Burchardt, Aljoscha, Vivien Macketanz, and Ankit Srivastava. DFKI's system for WMT16 IT-domain task, including analysis of systematic errors. In *Proceedings of the First Conference on Machine Translation*, pages 415–422, Berlin, Germany, aug 2016. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W/W16/W16-2329.

Banerjee, Somnath and Alon Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*. Association for Computational Linguistics, 2005.

Bojar, Ondřej, Rajen Chatterjee, Christian Federmann, Barry Haddow, Matthias Huck, Chris Hokamp, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Carolina Scarton, Lucia Specia, and Marco Turchi. Findings of the 2015 Workshop on Statistical Machine Translation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 1–46, Lisbon, Portugal, sep 2015. Association for Computational Linguistics. URL http://aclweb.org/anthology/W15-3001.

Brown, Peter F, Vincent J Della Pietra, Stephen A Della Pietra, and Robert L Mercer. The mathematics of statistical machine translation: parameter estimation. *Comput. Linguist.*, 19(2): 263–311, 1993. ISSN 0891-2017.

Callison-Burch, Chris, Cameron Fordyce, Philipp Koehn, Christof Monz, and Josh Schroeder. (Meta-) evaluation of machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation (StatMT'07)*, pages 136–158, Prague, Czech Republic, jun 2007. Association for Computational Linguistics. doi: 10.3115/1626355.1626373. URL http://www.statmt.org/wmt07/pdf/WMT18.pdf.

Callison-Burch, Chris, Cameron Fordyce, Philipp Koehn, Christof Monz, and Josh Schroeder. Further Meta-Evaluation of Machine Translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 70–106, Columbus, Ohio, jun 2008. Association for Computational Linguistics. URL www.aclweb.org/anthology/W08-0309.

Callison-Burch, Chris, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. Findings of the 2012 Workshop on Statistical Machine Translation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 10–51, Montréal, Canada, jun 2012. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W12-3102.

Cao, Zhe, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*, pages 129–136. ACM, 2007.

Demšar, Janez, Blaž Zupan, Gregor Leban, and Tomaz Curk. Orange: From Experimental Machine Learning to Interactive Data Mining. In *Principles of Data Mining and Knowledge Discovery*, pages 537–539, 2004. doi: 10.1007/b100704.

Heafield, Kenneth. KenLM : Faster and Smaller Language Model Queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, number 2009, pages 187–197, Edinburgh, Scotland, jul 2011. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W11-2123$\backslash$nhttp://kheafield.com/code/kenlm.

Koehn, Philipp, Wade Shen, Marcello Federico, Nicola Bertoldi, Chris Callison-Burch, Brooke Cowan, Chris Dyer, Hieu Hoang, Ondrej Bojar, Richard Zens, Alexandra Constantin, Evan Herbst, and Christine Moran. Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 177–180, Prague, Czech Republic, jun 2006.

Logacheva, Varvara, Chris Hokamp, and Lucia Specia. MARMOT: A Toolkit for Translation Quality Estimation at the Word Level. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France, may 2016. European Language Resources Association (ELRA). ISBN 978-2-9517408-9-1.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, jul 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135.

Pedregosa, F, G Varoquaux, A Gramfort, V Michel, B Thirion, O Grisel, M Blondel, P Prettenhofer, R Weiss, V Dubourg, J Vanderplas, A Passos, D Cournapeau, M Brucher, M Perrot, and E Duchesnay. Scikit-learn: Machine Learning in {P}ython. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Petrov, Slav, Leon Barrett, Romain Thibaux, and Dan Klein. Learning Accurate, Compact, and Interpretable Tree Annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440, Sydney, Australia, jul 2006. Association for Computational Linguistics.

Popović, Maja. Hjerson: An Open Source Tool for Automatic Error Classification of Machine Translation Output. *The Prague Bulletin of Mathematical Linguistics*, 96(-1):59–68, 2011. doi: 10.2478/v10108-011-0011-4.PBML. URL `www.mt-archive.info/PBML-2011-Popovic.pdf`.

Rückstieß, Thomas and Jürgen Schmidhuber. A Python Experiment Suite. *The Python Papers*, 6 (1):2, 2011.

Schmid, Helmut. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of International Conference on New Methods in Language Processing*, Manchester, UK, 1994.

Schmid, Helmut. Efficient Parsing of Highly Ambiguous Context-free Grammars with Bit Vectors. In *Proceedings of the 20th International Conference on Computational Linguistics*, COLING '04, Stroudsburg, PA, USA, 2004. Association for Computational Linguistics. doi: 10.3115/1220355.1220379. URL `http://dx.doi.org/10.3115/1220355.1220379`.

Servan, Christophe, Ngoc-Tien Le, Ngoc Quang Luong, Benjamin Lecouteux, and Laurent Besacier. An Open Source Toolkit for Word-level Confidence Estimation in Machine Translation. In *The 12th International Workshop on Spoken Language Translation (IWSLT'15)*, Da Nang, Vietnam, dec 2015. URL `https://hal.archives-ouvertes.fr/hal-01244477`.

Specia, Lucia, Kashif Shah, José Guilherme Camargo de Souza, and Trevor Cohn. QuEst - A translation quality estimation framework. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 79–84, Sofia, Bulgaria, aug 2013. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P13-4014`.

Specia, Lucia, Gustavo Paetzold, and Carolina Scarton. Multi-level Translation Quality Prediction with QuEst++. In *Proceedings of ACL-IJCNLP 2015 System Demonstrations*, pages 115–120, Beijing, China, jul 2015. Association for Computational Linguistics and The Asian Federation of Natural Language Processing. URL `http://www.aclweb.org/anthology/P15-4020`.

Tamchyna, Aleš, Ondřej Dušek, Rudolf Rosa, and Pavel Pecina. MTMonkey: A Scalable Infrastructure for a Machine Translation Web Service. *The Prague Bulletin of Mathematical Linguistics*, 100:31–40, oct 2013. ISSN 0032-6585. URL `http://ufal.mff.cuni.cz/pbml/100/art-tamchyna-dusek-rosa-pecina.pdf`.

van Cranenburgh, Andreas. Enriching Data-Oriented Parsing by blending morphology and syntax. Technical report, University of Amsterdam, Amsterdam, 2010. URL https://unstable.nl/andreas/ai/coglang/report.pdf.

van Cranenburgh, Andreas, Galit W Sassoon, and Raquel Fernández. Invented antonyms: Esperanto as a semantic lab. In *Proceedings of the 26th Annual Meeting of the Israel Association for Theoretical Linguistics (IATL 26).*, volume 26, 2010.

Wang, Wei and Knight, Kevin and Marcu, Daniel. Capitalizing machine translation. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conferenc*, pages 1–8, New York, 2006. URL http://dl.acm.org/citation.cfm?id=1220836.

Weissenborn, Dirk, Leonhard Hennig, Feiyu Xu, and Hans Uszkoreit. Multi-Objective Optimization for the Joint Disambiguation of Nouns and Named Entities. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*, pages 596–605, Beijing, China, 2015. Association for Computer Linguistics. ISBN 978-1-941643-72-3. URL http://aclweb.org/anthology/P/P15/P15-1058.pdf.

Address for correspondence:
Eleftherios Avramidis
eleftherios.avramidis@dfki.de
German Research Center for Artificial Intelligence (DFKI GmbH)
Language Technology Lab
Alt Moabit 91c
10559 Berlin, Germany