PBML

logo

The Prague Bulletin of Mathematical Linguistics
NUMBER ??? SEPTEMBER 2016 1–12

Language Adaptation for Extending Post-Editing Estimates for Closely Related Languages

Miguel Rios, Serge Sharoff

Centre for Translation Studies, University of Leeds

Abstract

This paper presents an open-source toolkit for predicting human post-editing efforts for closely related languages. At the moment, training resources for the Quality Estimation task are available for very few language directions and domains. Available resources can be expanded on the assumption that MT errors and the amount of post-editing required to correct them are comparable across related languages, even if the feature frequencies differ. In this paper we report a toolkit for achieving language adaptation, which is based on learning new feature representation using transfer learning methods. In particular, we report performance of a method based on Self-Taught Learning which adapts the English-Spanish pair to produce Quality Estimation models for translation from English into Portuguese, Italian and other Romance languages using the publicly available Autodesk dataset.

1. Introduction

A common problem with automatic metrics for Machine Translation (MT) evaluation, such as BLEU (Papineni et al., 2002), is the need to have reference human translations (Specia et al., 2010). Also such metrics work best on a corpus of segments, while they are not informative for evaluation of individual segments. The aim of Quality Estimation (QE) is to predict a quality score for a segment output by MT without its reference translation, for example, to predict Translation Edit Rate (TER), i.e., the distance between the raw MT output and its revised human output (Snover et al., 2006).

From the implementation viewpoint, the QE task can be framed as a regression problem aimed at predicting the amount of human TER, without the reference translations available. This helps in deciding whether an MT sentence can be a suitable

© 2016 PBML. Distributed under CC BY-NC-ND. Corresponding author: s.sharoff@leeds.ac.uk
Cite as: Miguel Rios, Serge Sharoff. Language Adaptation for Extending Post-Editing Estimates for Closely Related Languages. The Prague Bulletin of Mathematical Linguistics No. ???, 2016, pp. 1–12.
doi: HERE-WILL-BE-A-DOI-NUMBER

basis for human Post-Editing (PE) or it would be better to translate this sentence from scratch. The QE methods mostly rely on supervised Machine Learning (ML) algorithms aimed at computing similarity scores between a source sentence and its machine translations using a variety of sources of information, which are used as features to train a supervised ML algorithm to predict QE scores. Specia et al. (2013) developed QuEst, a baseline QE framework, which uses simple features quantifying the complexity of the source segment and its match to the machine translation output.

However, currently existing training datasets are only available for a limited number of languages. For example, in the WTM'15 QE task the available pairs were en-es and en-de,[1] which have been evaluated on the same domain (news). The end users of MT need a wider variety of language pairs and domains for evaluation. So far there has been little research to deal with this problem. Turchi and Negri (2014) proposed an automatic approach to produce training data for QE in order to tackle the problem of scarce training resources. Specia et al. (2010) used baseline QE framework across different domains and languages (i.e. en-es to en-dk). In our earlier work (Rios and Sharoff, 2015) we proposed using Transfer Learning (TL) for a training dataset from the WMT'14 QE task to predict PE labels, i.e., 'Perfect' vs 'Near miss' vs 'Low quality'.

In this paper, we describe the implementation of a transfer-based QE workflow to produce a large number of QE models for predicting the TER score by utilising the notion of relatedness between languages. More specifically, we use TL to learn better feature representations across related languages. Our intuition is that sentences with similar quality scores are near-neighbours in terms of QE features across related languages. In other words, good or bad quality sentences translated into Spanish (i.e., available training data) show similar characteristics to sentences translated into Portuguese (i.e., unlabelled data). This makes it possible to train a prediction algorithm by sharing information from the available labelled dataset with unlabelled datasets for related languages. However, to achieve reasonable prediction rate we need to adapt the feature representation for the dataset for the unlabelled language pair.

In this paper, we will present the Self-Taught Learning (STL) approach (Section 2), discuss the experimental setup (Section 3) and the implementation details of our toolkit (Section 3.3). We will also describe the dataset and analyse the results (Section 4).

2. Transfer Learning Methods

Transfer Learning aims to transfer information learned in one or more source tasks, i.e., using labelled datasets, to improve learning in a related target task without new annotations, i.e., using unlabelled datasets (Pan and Yang, 2010).

From the viewpoint of notation, the transfer models start with l labelled training examples $\{(x_1, y_1), (x_2, y_2)..., (x_l, y_l)\}$ and u unlabelled training examples $\{z_1, z_2, ..., z_u\}$.

[1] Throughout the paper we will be using the two-letter ISO codes to indicate the languages

The labels belong to a set of labels $\mathcal{Y}$ for the classification task or they are real-valued numbers for the regression task.

2.1. Self-Taught Learning

Raina et al. (2007) propose a semi-supervised transfer learning method, which does not assume that the unlabelled dataset is drawn from the same distribution as the labelled one. The unlabelled data is used to learn a lower dimensional representation of the input features. With this representation the labelled data can be used to learn a prediction model in the lower dimensional space, which better fits the unlabelled dataset.

Formally, the steps to perform STL are defined as:

1 Learn the dimensionality reduction for the unlabelled set z_i.
2 Compute a new representation for the labelled training dataset x_i.
3 Use standard classification/prediction methods with the new training dataset $f(\hat{x}_i) = y_i$.

The dimensionality reduction in our case is based on autoencoders. The autoencoder uses backpropagation to learn mapping the inputs to their own values via a hidden layer. The method learns an approximation function $h_{W,b}(z) \approx z$ similar the identity function, where W are the weights and b the bias. In Step 2, the labelled training data x_i is transformed by using the same parameters from the autoencoded unlabelled data $\hat{z}_i$. The new representation of the training data $\hat{x}_i$ is used to learn a prediction model in Step 3. The size of the lower-dimensional space is given by the number of units in the hidden layer. The STL model can be expanded to take into account several unknown signals, such as language pairs or domains.

Stacked autoencoders can perform a series of transformations of a labelled dataset given different autoencoders learned on several unlabelled datasets. In other words, each autoencoder is a layer L_n where the output of one autoencoder L_1 becomes the input of the following autoencoder L_2. For example, a two layer model has parameters $(W, b) = (W^1, b^1, W^2, b^2)$ for two stacked autoencoders.

3. Methodology

In this section, we describe the QE features and the transfer learning setup.

3.1. Features

The QE features come from information about the source sentence, its MT output and information about relations between them. The QuEst framework (Specia et al., 2013) implements 17 language-independent features classified into three types:

Complexity Indicators Features related to the difficulty in translating the source sentence, such as, the number of tokens of the source sentence, its language model and average number of translations in the phrase tables.

Fluency Indicators Features related to how fluent the MT output is, such as the language model of the target sentence.

Adequacy Indicators Features related to how much meaning is preserved in the MT output, such as, ratios of tokens between the source and target, ratio of punctuation and syntactic similarity. The QuEst framework also uses features related to a specific decoding process when available, such as, global score of the system and number of hypotheses in the n-best list.

In addition, we use a second set of features based on bilingual embeddings (Hermann and Blunsom, 2013), i.e., words and sentences from the source and target languages are positioned in a shared multidimensional representation, which assumes that words and sentences from one language are neighbours with words and sentences with similar meanings from another language. The motivation for introducing embeddings is to expand the range of Adequacy indicators using simple resources. The bilingual embeddings are induced from parallel data from the target domain. We build each sentence as an additive composition of individual word vectors. The final vector is a concatenation of vectors from the source sentence and its machine translation. The final embedding vector for the experiments consists of 200 features.

3.2. Implementation Details

Texts in related languages are treated as unlabelled data. For example, the available en-es labelled dataset is used to transfer information into the unlabelled en-pt sentences to predict their QE scores. We compare the transfer-based QE workflow that uses Self-Taught Learning (STL) against the Baseline with no transfer. There developed workflows can tackle both QE scenarios: the prediction of HTER and the classification of post-editing effort.

For all the HTER prediction workflows we use the Support Vector Regression (SVR) algorithm with the RBF kernel from scikit-learn (Pedregosa et al., 2011). The hyperparameters C and ϵ have been determined analytically following (Cherkassky and Ma, 2004): $\epsilon = 3\sigma(y)\sqrt{\ln(n)/n}$ and $C = mean(y)+3\sigma(y)$ where y is HTER in the training set, σ is the standard deviation, n is the number of observations.

For STL we modified the autoencoder implementation from Theano (Bergstra et al., 2010). The STL model first finds the weights W, b from the unlabelled z_i dataset by training a sparse autoencoder. Second, the model produces a modified training dataset by using the unlabelled weights on an second autoencoder. The modified training dataset is a lower-dimensional representation of the input features. A new test dataset can be predicted by using the weights W, b to represent the data points into the same lower-dimensional space. However, we do not have access to any development datasets for tuning the z_i autoencoder for our unlabelled language pairs. For the parameter selection of the unlabelled autoencoder, as suggested in Bergstra and Bengio (2012), we run a random search over a split of the modified training dataset (90% training, 10% validation) in order to find: the size of the hidden dimension, the

desired average activation sparsity parameter (ρ), the weight decay parameter (λ) and the sparsity penalty (β).

The stacked STL setup can be used for language pairs where the source is different form the available training dataset. For example, the training dataset is en-es and the objective test is fr-es. The first autoencoder is trained with en-fr and the second autoencoder with fr-es, which projects training en-es first into the space of en-fr, and then into fr-es.

In addition to STL, we also experimented with other TL strategies, namely multi-view learning and Transductive SVM. The multi-view learning framework tries to jointly optimise different views of the same input (Xu et al., 2013). Spectral methods such as Canonical Correlation Analysis (CCA) can be used to learn a subspace shared by label and unlabelled data. The Spectral method is straightforward to apply to two-view data. In our case, the first view is the labelled data x_i and the second view is the unlabelled data z_i. CCA learns two projections $A_m \epsilon \mathbb{R}^{l \times m}$ where l are the labelled instances and m the number of features, and $B_m \epsilon \mathbb{R}^{u \times m}$. We use A_m to project each instance of the test dataset into $\hat{x}_i$. For the Spectral Learning setup, we use the CCA implementation from MATLAB[2] and the same SVR setup as for STL. For example, the available dataset is en-es and the test objective en-pt. We use CCA to learn the projections of en-es and en-pt. The en-es test is projected into the same lower space with A_i, and then, we use the projected datasets for training and testing respectively.

The methods described above can be used in different QE scenarios by changing from SVR to SVM. In particular for the classification of post-editing effort, Transductive Support Vector Machine (TSVM) takes into consideration a particular test dataset and tries to minimise errors only on those particular instances (Vapnik, 1995). The TSVM model learns a large margin hyperplane classifier using labelled training data, but at the same time it forces that hyperplane to be far from the unlabelled data, and the method transfers the information from labelled instances to the unlabelled. We use SVMlin[3] for training the TSVM. TSVM uses an Linear kernel with no hyper-parameter optimisation. We select the heuristic Multi-switch TSVM. Each instance in the unlabelled dataset is added to the training dataset. For classification, we implement the one-against-one strategy, and the final decision is given by voting.

The standard QE baseline measures HTER prediction without any adaptation, i.e., the en-es QE prediction model is applied to en-pt data.

For the regression scenario, we report the Mean Absolute Error (MAE), Root Mean Squared Error (RSME) and Pearson correlation. Our main evaluation metric is the Pearson correlation as suggested in Graham (2015).

[2] http://uk.mathworks.com/help/stats/canoncorr.html

[3] http://vikas.sindhwani.org/

3.3. QE Transfer Command Usage

In this section, we show the different implemented methods for transfer-based QE. The repository contains the scripts for extracting the features and implementations of transfer-based QE methods, where each transfer workflow uses the same input parameters. The first step of the transfer-based workflow is to extract features for: the labelled dataset, the unlabelled and test datasets. For the feature representation, we have available two feature extractor scripts. The QuEst feature extractor that depends on QuEst[4] and Moses. The bilingual embeddings feature extractor that depends on BICVM[5].

The next step is to train and predict the test dataset. We developed different QE adaptation tools based on transfer-learning such as: STL, stacked STL, CCA all for regression and classification, and TSVMy for classification. The final and optional step is to measure the predicted HTER against a gold-standard annotation of the test dataset.

In addition, we implemented the analytical method to estimate the parameters ϵ and C of the SVR, where the input is the training examples features.

Preliminary results show that STL outperforms other transfer learning methods over both regression and classification. We show the use of the STL transfer method given the QuEst baseline features. The input parameters of the adapted QE based on STL with SVR for HTER prediction are: (1) training examples features, (2) training labels, (3) unlabelled training examples features, (4) test features, (5) output, (6) epsilon parameter for SVR, (7) C parameter for SVR and (8) size of hidden layer for the autoencoder. Parameters (6)-(8) will be determined as discussed above if not provided explicitly. An example of the command is as follows:

```
python stlSVR.py \
--training-examples autodesk.training.en-es.feat \
--training-labels autodesk.training.en-es.hter \
--unlabelled-examples autodesk.training.en-pt.feat \
--test autodesk.test.en-pt.feat \
--output autodesk.en-pt.pred \
--epsilon 41.06 \
--c 0.232 \
--hidden-layer 50
```

The default parameters for the autoencoder have been selected via random search over a split on the labelled language dataset. It is worth noticing that we do not constraint the number of hidden units during the learning of the autoencoder. Thus, we

[4] http://www.quest.dcs.shef.ac.uk/

[5] https://github.com/karlmoritz/bicvm

set a bound for random search from 0 to 100 units, and for our example the optimum number of units has been detected as *50*.

4. Experiments

In this section, we describe the datasets used to train and evaluate our transfer learning model for pairs of related languages. We show the results of the STL transfer-based QE and we also discuss the predicted scores for different language pairs.

4.1. Dataset

In this paper, we experimented with the Autodesk PE dataset (Zhechev, 2012).[6] The Autodesk corpus contains the source, MT and PE segments for several languages. The corpus consist of user manuals, marketing and educational material for the Autodesk applications, such as AutoCAD, REVIT, Inventor. We use the following language pairs showed in Table 1, with a 70/30% split for the training/test data.

Language Pair	Training Labelled	Training Unlabelled	Test
en-es	24,073	-	8,025
en-pt	-	28,886	9,629
en-it	-	30,311	10,104
en-fr	-	38,469	12,824
en-ru	30,905	-	10,302
en-cs	-	20,997	7,000
en-pl	-	24,853	8,285
fr-es	-	10,000	1,000
it-es	-	10,000	1,000
pt-es	-	10,000	1,000

Table 1. Autodesk training and test number of segments used in this study.

We use as labelled training data *en-es* for the Romance family and *en-ru* for the Slavonic family. The remaining language pairs were used as unlabelled and test data for each family. Given that the Czech dataset is much smaller, it has been only used for tuning/testing. The unlabelled set has been produced by running the remaining English segments **not included** in the en-cs set through Google MT.

The QE score (HTER) is the minimum number of edit operations (TER) between the MT output and PE. We use Tercom (Snover et al., 2006) to compute the HTER scores between the post-edited and MT segments.

[6] `https://autodesk.app.box.com/v/autodesk-postediting`

We produce the pt-es, it-es and fr-es language pairs by intersecting the English segments. For example, the same English segments present in en-pt and en-es produces the pt-es alignment for both MT and PE. For extracting the QuEst features, we use Moses (Koehn et al., 2007) and KenLM (Heafield, 2011) with a 3-gram language model (LM).

4.2. Results

In this section, we show the results of our proposed STL QE workflow against the standard QE Baseline. We built the transfer-based QE and baseline models for the language directions in Table 2.

Training labelled	Test unlabelled
en-es	en-pt, en-it, en-fr
	pt-es, it-es, fr-es
en-ru	en-cs, en-pl

Table 2. Language directions workflows.

The upper bound for our TL methods is the standard QE setup in which the same feature set is used for training and testing on the same language pair, en-es and en-ru in our case (Table 3).

Training en-es		en-es
Upper baseline	MAE	0.14
	RSME	0.18
	Pearson	**0.53**
Training en-ru		en-ru
Upper baseline	MAE	0.18
	RSME	0.27
	Pearson	**0.47**

Table 3. Upper-bound baseline for labelled language pairs.

Table 4 shows the transfer results for the workflows. Over the Romance pair we can see consistent and considerable improvement over the baseline with no adaptation, e.g., $0.35 \rightarrow 0.52$ for correlation in the case of en-es→en-pt TL, which approaches the

Training en-es		en-pt	en-it	en-fr
STL	MAE	0.14	0.16	0.17
	RMSE	0.17	0.21	0.22
	Pearson	**0.52**	**0.40**	**0.30**
Baseline	MAE	0.16	0.18	0.18
	RMSE	0.20	0.23	0.23
	Pearson	0.35	0.26	0.24

Training en-ru		en-cs	en-pl
STL	MAE	0.19	0.19
	RMSE	0.25	0.25
	Pearson	**0.41**	**0.46**
Baseline	MAE	0.20	0.21
	RMSE	0.26	0.27
	Pearson	0.32	0.33

Table 4. Transfer learning results.

upper baseline of 0.53 for training and testing on the same language pair (en-es). For the Slavonic language pairs we also reach the upper baseline for the en-pl pair.

Training en-es		pt-es	it-es	fr-es
STL	MAE	0.18	0.18	0.18
	RSME	0.23	0.22	0.22
	Pearson	**0.19**	**0.23**	**0.21**
Stacked STL	MAE	0.20	0.58	0.24
	RMSE	0.25	0.62	0.30
	Pearson	0.07	0.06	0.02
Baseline	MAE	0.19	0.19	0.18
	RSME	0.23	0.24	0.22
	Pearson	0.14	0.17	0.10

Table 5. Transfer learning results with en-es training into test: pt-es, it-es and fr-es.

Table 5 shows the transfer results **across** the Romance language pairs. The training is en-es and we adapt to pt-es, it-es and fr-es.

Table 6 shows the transfer-based results and the Baseline for comparison between more distant languages. As expected, the performance of TL is much lower between non related languages, i.e., no useful adaptation is taking place.

Training en-es		en-cs	en-pl
STL	MAE	0.22	0.25
	RMSE	0.29	0.32
	Pearson	*0.08*	*0.11*
Baseline	MAE	0.23	0.22
	RSME	0.31	0.29
	Pearson	*0.11*	*0.09*

Table 6. Transfer learning results with en-es training into test: en-cs and en-pl.

The features of the source and target directions affect the results of the transfer methods, i.e. complexity, fluency and adequacy indicators. For example, in the case of STL adaptation from en-es to pt-es, there is no agreement between the features of the source languages (en vs pt, complexity indicators) given they are not closely related, but the target languages are closely related. However, when the source languages are the same (en-es $\rightarrow$ en-pt) and the target languages are closely related, i.e. the fluency indicators can be transformed, the overall performance improves nearly up to the level of the labelled (upper-bound) pair baseline.

In addition to RMSE and correlation scores, there is a danger that adaptation can produce a narrow range of predicted values in comparison to the test set. We analyse the results of transfer by presenting the range of HTER predicted scores at (10%, 90%) quantiles, i.e. by trimming 10% of the most extreme values, which are likely to contain the outliers.

The (10%, 90%) quantile range for en-es $\rightarrow$ en-pt is as follows: Gold (0.0, 0.53), STL (0.23, 0.46) and Baseline(0.16, 0.47). The spread of the STL predicted values is slightly less than the baseline. For the stacked STL a possible reason for the low performance is related to over-fitting. The range for en-es $\rightarrow$ pt-es is: Gold (0.0, 0.57) and Stacked STL (0.50, 0.50). A better configuration of en-es $\rightarrow$ pt-es with the the stacked STL can be: en-es (training), es-pt (first layer) and pt-es (second layer). The motivation is to find an agreement between the source and the target features with the addition of more closely languages in terms of the induced lower dimensional space.

5. Conclusions and Future Work

We present an open-source toolkit[7] for transferring QE features from a single training dataset to closely related languages via Self-Taught Learning. We also developed other transfer learning methods for the task of QE prediction. It has been found successful in prediction the PE operations on the Autodesk dataset for the Romance and Slavonic families. For the reasons of testing the method the language pairs involved in

[7] `https://github.com/mriosb08/palodiem-QE`

the experiment do have suitable training resources. However, such sizeable datasets are rare. Even the Autodesk set only covers three Slavonic languages, while only German is available for the Germanic languages in this set.

One possibility for further research concerns the expansion of the available labelled resources with adaptation to different *domains* in addition to the language families, for example, by transferring predictions from the original domain of the Autodesk PE set to other domains with only unlabelled data available.

Acknowledgements

This study was funded as a research grant by Innovate UK and ZOO Digital.

Bibliography

Bergstra, James and Yoshua Bengio. Random Search for Hyper-parameter Optimization. *J. Mach. Learn. Res.*, 13:281–305, Feb. 2012. ISSN 1532-4435.

Bergstra, James, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. Theano: a CPU and GPU Math Expression Compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, June 2010. Oral Presentation.

Cherkassky, Vladimir and Yunqian Ma. Practical selection of SVM parameters and noise estimation for SVM regression. *Neural networks*, 17(1):113–126, 2004.

Graham, Yvette. Improving Evaluation of Machine Translation Quality Estimation. 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, ACL 2015, July 26-31, 2015, Beijing*, 2015.

Heafield, Kenneth. KenLM: Faster and Smaller Language Model Queries. In *Proceedings of the EMNLP 2011 Sixth Workshop on Statistical Machine Translation*, pages 187–197, Edinburgh, Scotland, United Kingdom, July 2011.

Hermann, Karl Moritz and Phil Blunsom. A Simple Model for Learning Multilingual Compositional Semantics. *CoRR*, abs/1312.6173, 2013.

Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 177–180, Stroudsburg, PA, USA, 2007.

Pan, Sinno Jialin and Qiang Yang. A Survey on Transfer Learning. *IEEE Trans. on Knowl. and Data Eng.*, 22(10):1345–1359, Oct. 2010.

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 on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA, 2002.

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 Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Raina, Rajat, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y. Ng. Self-taught Learning: Transfer Learning from Unlabeled Data. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pages 759–766, New York, NY, USA, 2007. ISBN 978-1-59593-793-3.

Rios, Miguel and Serge Sharoff. Large Scale Translation Quality Estimation. In *The Proceedings of the 1st Deep Machine Translation Workshop*, Praha, Czech Republic, 2015.

Snover, Matthew, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of Association for Machine Translation in the Americas,*, 2006.

Specia, Lucia, Dhwaj Raj, and Marco Turchi. Machine translation evaluation versus quality estimation. *Machine Translation*, 24(1):39–50, 2010.

Specia, Lucia, Kashif Shah, Jose G.C. de Souza, and Trevor Cohn. QuEst - A translation quality estimation framework. In *51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, ACL, pages 79–84, Sofia, Bulgaria, 2013.

Turchi, Marco and Matteo Negri. Automatic Annotation of Machine Translation Datasets with Binary Quality Judgements. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland, may 2014.

Vapnik, Vladimir N. *The Nature of Statistical Learning Theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995. ISBN 0-387-94559-8.

Xu, Chang, Dacheng Tao, and Chao Xu. A Survey on Multi-view Learning. *CoRR*, abs/1304.5634, 2013. URL `http://arxiv.org/abs/1304.5634`.

Zhechev, Ventsislav. Machine Translation Infrastructure and Post-editing Performance at Autodesk. In *Proc AMTA*, San Diego, CA, 2012.

Address for correspondence:
Serge Sharoff
`s.sharoff@leeds.ac.uk`
Centre for Translation Studies,
Parkinson Bldg, University of Leeds
LS2 9JT, UK