TranscRater: a Tool for Automatic Speech Recognition Quality Estimation

Shahab Jalalvand$^{(1,2)}$, **Matteo Negri**$^{(1)}$, **Marco Turchi**$^{(1)}$,
José G. C. de Souza$^{(1,2)}$, **Daniele Falavigna**$^{(1)}$, **Mohammed R. H. Qwaider**$^{(1)}$
$^{(1)}$ Fondazione Bruno Kessler, Trento, Italy
$^{(2)}$ University of Trento, Italy
{jalalvand,negri,turchi,desouza,falavi,qwaider}@fbk.eu

Abstract

We present TranscRater, an open-source tool for automatic speech recognition (ASR) quality estimation (QE). The tool allows users to perform ASR evaluation bypassing the need of reference transcripts and confidence information, which is common to current assessment protocols. TranscRater includes: *i)* methods to extract a variety of quality indicators from (*signal, transcription*) pairs and *ii)* machine learning algorithms which make possible to build ASR QE models exploiting the extracted features. Confirming the positive results of previous evaluations, new experiments with TranscRater indicate its effectiveness both in WER prediction and transcription ranking tasks.

1 Introduction

How to determine the quality of an automatic transcription without reference transcripts and without confidence information? This is the key problem addressed by research on ASR quality estimation (Negri et al., 2014; C. de Souza et al., 2015; Jalalvand et al., 2015b), and the task for which TranscRater, the tool described in this paper, has been designed.

The work on ASR quality estimation (ASR QE) has several motivations. First, the steady increase of applications involving automatic speech recognition (e.g. video/TV programs subtitling, voice search engines, voice question answering, spoken dialog systems, meeting and broadcast news transcriptions) calls for **an accurate method to estimate ASR output quality at run-time**. Often, indeed, the nature of such applications (consider for instance spoken dialog systems) requires quick response capabilities that are incompatible with traditional reference-based protocols.

Second, even when real-time processing is not a priority, standard evaluation based on computing word-error rate (WER) against gold references is not always a viable solution. In many situations (as in the case of languages for which even the ASR training data is scarce), the bottleneck represented by the limited availability of reference transcripts and the costs of their manual creation calls for **a method to predict ASR output quality that is reference-independent**.

Third, even when designed to bypass the need of references, current quality prediction methods heavily depend on confidence information about the inner workings of the ASR system that produced the transcriptions (Evermann and Woodland, 2000; Wessel et al., 2001). Such information, describing how the system is certain about the quality of its own hypotheses, often reflects a biased perspective influenced by individual decoder features. More importantly, it is not always accessible and, in this frequent case, the sole elements available for quality prediction are the signal and its transcription (consider, for instance, the increasing amount of captioned Youtube videos generated by a "black-box" ASR system[1]). These issues call for **a method to predict ASR output quality that is also confidence-independent**.

TranscRater (Transcription Rater) provides a unified ASR QE framework designed to meet the three aforementioned requirements. Its development was inspired by software previously released for the machine translation (MT) (Specia et al., 2013; Shah et al., 2014; Servan et al., 2015) equivalent of ASR QE, in which MT quality has to be estimated at run-time and without reference trans-

[1]More than 157 millions in 10 languages, as announced by Google already in 2012 (source: http://goo.gl/5Wlkjl).

lations (Mehdad et al., 2012; Camargo de Souza et al., 2013; C. de Souza et al., 2014). Indeed, the two tasks deal with similar issues. In both cases, we have an input "source" (a written sentence and a recorded signal) and an output text (a translation and a transcription) that has to be assessed without any pre-created term of comparison. They can also be approached with similar supervised classification (C. de Souza et al., 2015) or regression strategies (Negri et al., 2014; C. de Souza et al., 2015). Finally, they have similar applications like:

- Deciding if an input source has been correctly processed;
- Ranking the output of multiple independent systems (Jalalvand et al., 2015b);
- Estimating the human effort required to manually revise an output segment;
- Performing data selection for system improvement based on active learning.

To support these applications, TranscRater provides an extensible ASR QE framework consisting of a variety of feature extractors and machine learning algorithms. The implemented feature extraction methods allow capturing predictive quality indicators both from the input signal and from the output transcription. This basic set of "black box" indicators has been successfully evaluated in a number of experiments, both on regression and on classification tasks, showing that ASR QE predictions can closely approximate the quality scores obtained with standard reference-based methods. The existing feature extractors can be easily extended to integrate new features, either capturing additional system-independent aspects, or relying on confidence information about the ASR system that produced the transcriptions, if available. Experimental results demonstrate that, also in the "glass-box" scenario in which the ASR system is known, the available features are able to improve the performance obtained with confidence information.

The integration of different machine learning algorithms makes TranscRater a powerful framework to quickly set up an ASR QE model given some training data, tune it by choosing among the possible feature configurations and process new, unseen test data to predict their quality. As a standalone environment, with few documented external dependencies, TranscRater provides the first off-the-shelf solution to approach ASR QE and extend its application to new scenarios.

2 ASR QE

The basic ASR QE task consists in training a model from (*signal, transcription, label*) triplets, and using it to return quality predictions for a test set of unseen (*signal, transcription*) instances. In this supervised learning setting, the training labels can be either numeric scores (Negri et al., 2014) or class identifiers (binary or multi-class) (C. de Souza et al., 2015). Class assignments can be manually done according to some criteria, or inferred by thresholding numeric scores. Numeric quality indicators can be easily obtained by measuring the similarity (or the distance) between the transcription and its manually-created reference. For instance, the models described in previous works on ASR QE learn from training data labelled with real values obtained by computing the transcription word error rate (WER[2]).

According to the type of training labels, the problem can be approached either as a regression or as a classification task. As a consequence, also the evaluation metrics will change. Precision/recall/F1 (or other metrics, such as balanced accuracy, in case of very unbalanced distributions) will be used for classification while, similar to MT QE, the mean absolute error (MAE) or similar metrics will be used for regression.

A variant of the basic ASR QE task is to consider it as a QE-based ranking problem (Jalalvand et al., 2015b), in which each utterance is captured by multiple microphones or transcribed by multiple ASR systems. In this case, the capability to rank transcriptions from the best to the worst can be evaluated in terms of normalized discounted cumulative gain (NDCG) or similar metrics.

3 The TranscRater tool

TranscRater combines in a single open-source framework: *i)* a set of *features* capturing different aspects of transcription quality and *ii)* different learning *algorithms* suitable to address the challenges posed by different application scenarios.

TranscRater internally consists of two main

[2]WER is the minimum edit distance between the transcription and the reference. Edit distance is calculated as the number of edits (word insertions, deletions, substitutions) divided by the number of words in the reference.

modules: feature extraction and machine learning. At training stage, the tool receives as input a set of signal recordings, their transcriptions and the corresponding reference transcripts. The speech signals are provided as separate files in the RIFF Microsoft PCM format with 16K sampling rate. Their transcriptions and the corresponding references are provided in single separate text files (one transcription per row). References are used to compute the WER label of each training instance, thus connecting the problem to the task formulation provided in §2. The features extracted from each training instance are passed to the learning module, together with the corresponding label. The label is a WER score which, depending on the type of problem addressed, can be used either to directly train a regressor or to infer a ranking for multiple hypotheses. In either case, the learning module will train the corresponding model with the proper learning algorithm, and tune it using k-fold cross-validation.

At test stage, the model is used to predict the label of new, unseen (*signal*, *transcription*) instances. For each test point, the output is either a WER prediction or a rank, whose reliability can be respectively evaluated in terms of MAE or NDCG (as discussed in §2). Output predictions are provided in a single file (one WER prediction per row for regression and one rank prediction per row for ranking). MAE or NDCG scores are provided as the standard output of the test functions.

Internally, TranscRater stores the extracted features in the SVM-light[3] format. This makes possible to use the tool as a feature extractor and to embed it in applications different from the ones described in this paper. The features to be used, the type of learning algorithm, the input files and the links to resources and libraries can be easily set through a configuration file.

3.1 Feature sets

The feature extraction module of TranscRater allows the user to extract 72 features that can be categorized in the following four groups:

- Signal (SIG) features, designed to capture the difficulty of transcribing the input signal given the general recording conditions in which it was acquired;
- Lexical (LEX) features, designed to capture the difficulty to transcribe the input signal given the pronunciation difficulty and the ambiguity of the terms it contains;
- Language model (LM) features, designed to capture the plausibility of the transcription from the fluency point of view;
- Part-of-speech (POS) features, designed to capture the plausibility of the transcription from the syntax point of view.

SIG (44). Signal features are extracted using the OpenSmile[4] toolkit (Eyben et al., 2013). Each speech signal is broken down into 25ms length frames with 10ms overlap. For each frame, we compute 13 Mel Frequency Cepstral Coefficients (MFCC), their delta, acceleration and log-energy as well as the prosody features like fundamental frequency (F0), voicing probability, loudness contours and pitch. The final SIG feature vector for the entire input signal is obtained by averaging the values of each feature computed on all the frames.

LEX (7). Lexicon-based features are extracted using a lexical feature dictionary (optionally provided by the user). In this dictionary each individual word is assigned to a feature vector containing the frequency of fricatives, liquids, nasals, stops and vowels in its pronunciation. Other elements of the vector are the number of homophones (words with the same pronunciation) and quasi-homophones (words with similar pronunciation).

LM (12). Language model features include the mean of word probabilities, the sum of the log probabilities and the perplexity score for each transcription. In previous experiments (Jalalvand et al., 2015b; Jalalvand and Falavigna, 2015) we showed that, instead of only one LM, using a combination of neural network and n-gram LMs trained on task-specific and generic data can significantly improve the accuracy of quality prediction. For this reason, TranscRater allows using up to four different language models: two RNNLM (Mikolov et al., 2010) trained on generic and specific data and two n-gramLM trained on generic and specific data. To work with neural network LMs, the tool makes use of RNNLM,[5] while for n-gram LMs it uses SRILM[6] (Stolcke et al., 2000).

[3]http://svmlight.joachims.org/

[4]http://www.audeering.com/research/opensmile\#download

[5]http://www.fit.vutbr.cz/~imikolov/rnnlm/rnnlm-0.3e.tgz

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

POS (9). Part-of-speech features are extracted using the TreeTagger.[7] For each word in the transcription, they consider the score assigned to the predicted POS of the word itself, the previous and the following one. This sliding window is used to compute the average value for the entire transcription and obtain the sentence-level POS feature vector. The intuition is that a low confidence of the POS tagger in labeling a sentence is an indicator of possible syntax issues and, in turn, of poor transcription quality. POS features also include the number and the percentage of content words (numbers, nouns, verbs, adjectives, adverbs).

These feature groups were successfully tested in various conditions including clean/noisy data, single/multiple microphones and ASR systems (Jalalvand et al., 2015b; Jalalvand et al., 2015a). In such conditions, they proved to be a reliable predictor when confidence information about the ASR system inner workings is not accessible.

3.2 Learning algorithms

For regression-based tasks (WER prediction), TranscRater includes an interface to the Scikit-learn package (Pedregosa et al., 2011), a Python machine learning library that contains a large set of classification and regression algorithms. Based on the empirical results reported in (Negri et al., 2014; C. de Souza et al., 2015; Jalalvand et al., 2015b), which indicate that Extremely Randomized Trees (XRT (Geurts et al., 2006)) is a very competitive algorithm in several WER prediction tasks, the current version of the tool exploits XRT. However, adapting the interface to apply other algorithms is an easy task and one of the future extension directions. The main hyper-parameters of the model, such as the number of tree bags, the number of trees per bag, the number of features per tree and the number of instances in the leaves, are tuned using grid search with k-fold cross-validation on the training set to minimize the mean absolute error (MAE) between the true WERs and the predicted ones.

As mentioned before, TranscRater provides the possibility to evaluate multiple transcriptions (e.g. obtained from different microphones or ASR systems) and rank them based on their quality. This can be done either *indirectly*, by exploiting the predicted WER labels in a "ranking by regression" approach (RR) or *directly*, by exploiting machine-learned ranking methods (MLR). To train and test MLR models, TranscRater exploits RankLib[8], a library of learning-to-rank algorithms. The current version of the tool includes an interface to the Random Forest algorithm (RF (Breiman, 2001)), the same used in (Jalalvand et al., 2015b).

MLR predicts ranks through pairwise comparison between the transcriptions. The main parameters such as the number of bags, the number of trees per bag and the number of leaves per tree are tuned on training set using k-fold cross-validation to maximize the NDCG measure.

3.3 Implementation

TranscRater is written in Python and is made of several parts linked together using bash scripts. In order to run the toolkit on Linux, the following libraries are required: *i*) Java 8 (JDK-1.8); *ii*) Python 2.7 (or above) and *iii*) Scikit-learn (version 0.15.2). Moreover, the user has to download and compile the following libraries: OpenSmile, RNNLM, SRILM and TreeTagger for the feature extraction module as well as RankLib for using machine-learned ranking option.

4 Benchmarking

The features and algorithms contained in TranscRater have been successfully used in previous works (Negri et al., 2014; C. de Souza et al., 2015; Jalalvand et al., 2015b; Jalalvand et al., 2015a). To further investigate their effectiveness, in this section we provide new results, both in WER prediction (MAE) and transcription ranking (NDCG), together with some efficiency analysis (Time in seconds[9]). To this aim, we use data from the 3rd CHiME challenge,[10] which were collected for multiple distant microphone speech recognition in noisy environments (Barker et al., 2015). CHiME-3 data consists of sentences of the Wall Street Journal corpus, uttered by four speakers in four noisy environments, and recorded by five microphones placed on the frame of a tablet PC (a sixth one, placed on the back, mainly records background noise). Training and test respectively contain 1,640 and 1,320 sentences. Transcriptions are

[7] `http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/data/tree-tagger-linux-3.2.tar.gz`

[8] `https://people.cs.umass.edu/~vdang/data/RankLib-v2.1.tar.gz`

[9] Experiments were run with a PC with 8 Intel Xeon processors 3.4 GHz and 8 GB RAM.

[10] `http://spandh.dcs.shef.ac.uk/chime_challenge/chime2015/data.html`

produced by a baseline ASR system, provided by the task organizers, which uses the deep neural network recipe of Kaldi (Povey et al., 2011).

In WER prediction, different models built with TranscRater are compared with a baseline commonly used for regression tasks, which labels all the test instances with the average WER value computed on the training set. In ranking mode, baseline results are computed by averaging the NDCG scores obtained in one hundred iterations in which test instances are randomly ranked.

Features	Train&Test Time	Total Time	MAE↓
Baseline	—	—	28.7
SIG	00m18s	09m32s	27.3
LEX+LM+POS	00m19s	01m19s	22.2
SIG+LEX+LM+POS	00m26s	10m22s	23.5

Table 1: Time and MAE results in regression mode.

Table 1 shows the results of models trained with different feature groups for **WER prediction** with a single microphone. In terms of time, in this as in the following experiments, the total time (feature extraction + training + test) is mostly determined by feature extraction and the bottleneck is clearly represented by the extraction of signal (SIG) features. In terms of MAE, SIG features are also those achieving the worst result. Although they significantly improve over the baseline, they are outperformed by LEX+LM+POS and, even in combination with them, they do not help. However, as suggested by previous works like (Negri et al., 2014) in which some of the SIG features are among the most predictive ones, the usefulness of signal features highly depends on data and, in specific conditions, they definitely improve results. Their ineffectiveness in the experiments of this paper likely depends on the lack of word-level time boundaries, which prevented us to compute more discriminative features like word log-energies, noise log-energies and signal-to-noise ratio (the best indicator of the acoustic quality of an input utterance).

Features	Train&Test Time	Total Time	NDCG↑
Baseline	—	—	73.6
SIG	02m03s	15m11s	73.5
LEX+LM+POS	01m10s	03m13s	80.4
SIG+LEX+LM+POS	05m53s	19m23s	79.4

Table 2: Time and NDCG results in ranking by regression.

Table 2 shows the results achieved by the same feature groups when **ranking by regression** (RR) the transcriptions from five microphones. In terms of computation time, the higher costs of SIG features are still evident (the significant increase for all groups is due to the higher number of audio files to be processed). Also in this case, SIG features do not help, neither alone nor in combination with the other groups. Indeed, the highest results are achieved by the combination of LEX+LM+POS. Their large NDCG improvement over the baseline (+6.8), combined with the significantly lower computation time, seems to make this combination particularly suitable for the ranking by regression strategy.

Features	Train&Test Time	Total Time	NDCG↑
Baseline	—	—	73.6
SIG	01m14s	13m00s	78.1
LEX+LM+POS	00m59s	03m05s	81.3
SIG+LEX+LM+POS	01m41s	15m10s	83.1

Table 3: Time and NDCG with machine-learned ranking.

Table 3 shows the results achieved, in the same multi-microphone scenario, by the **machine-learned ranking** approach (MLR). In terms of time, MLR is slightly more efficient than RR, at least on this dataset. Though surprising (MLR performs lots of pairwise comparisons, which are in principle more demanding), such difference is not very informative as it might depend on hyper-parameter settings (e.g. the number of iterations for XRT, manually set to 20), whose optimization was out of the scope of our analysis. In terms of NDCG, the results are higher compared to RR but the differences between feature groups are confirmed. Interestingly, with MLR even the SIG features in isolation significantly improve over the baseline (+4.5 points). The NDCG improvement with the combined feature groups is up to 9.5 points, confirming the effectiveness of the combined features shown in previous works.

5 Conclusion

We presented TranscRater, an open-source tool for ASR quality estimation. TranscRater provides an extensible framework including feature extractors, machine learning algorithms (for WER prediction and transcription ranking), optimization and evaluation functions. Its source code can be downloaded from `https://github.com/hlt-mt/TranscRater`. Its license is FreeBSD, a lax permissive non-copyleft license, compatible with the GNU GPL and with any use, including commercial.

References

J. Barker, R. Marxer, E. Vincent, and S. Watanabe. 2015. The third 'CHiME' Speech Separation and Recognition Challenge: Dataset, Task and Baselines. In *Proc. of the 15th IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 1–9, Scottsdale, Arizona, USA.

L. Breiman. 2001. Random Forests. *Machine Learning*, 45(1):5–32.

J. G. C. de Souza, J. González-Rubio, C. Buck, M. Turchi, and M. Negri. 2014. FBK-UPV-UEdin participation in the WMT14 Quality Estimation shared-task. In *Proc. of the Ninth Workshop on Statistical Machine Translation*, pages 322–328, Baltimore, Maryland, USA.

J. G. C. de Souza, H. Zamani, M. Negri, M. Turchi, and D. Falavigna. 2015. Multitask Learning for Adaptive Quality Estimation of Automatically Transcribed Utterances. In *Proc. of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies (NAACL-HLT)*, pages 714–724, Denver, Colorado, USA.

J. G. Camargo de Souza, C. Buck, M. Turchi, and M. Negri. 2013. FBK-UEdin Participation to the WMT13 Quality Estimation Shared Task. In *Proc. of the Eighth Workshop on Statistical Machine Translation*, pages 352–358, Sofia, Bulgaria.

G. Evermann and P. Woodland. 2000. Large Vocabulary Decoding and Confidence Estimation using Word Posterior Probabilities. In *Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1655–1658, Istanbul, Turkey.

F. Eyben, F. Weninger, and B. Schuller. 2013. Recent developments in opensmile, the munich open-source multimedia feature extractor. In *Proc. of ACM Multimedia*, pages 835–838, Barcelona, Spain. ACM.

P. Geurts, D. Ernst, and L. Wehenkel. 2006. Extremely Randomized Trees. *Machine Learning*, 63(1):3–42.

S. Jalalvand and D. Falavigna. 2015. Stacked Auto-Encoder for ASR Error Detection and Word Error Rate Prediction. In *Proc. of the 16th Annual Conference of the International Speech Communication Association (INTERPSEECH)*, pages 2142–2146, Dresden, Germany.

S. Jalalvand, D. Falavigna, M. Matassoni, P. Svaizer, and M. Omologo. 2015a. Boosted Acoustic Model Learning and Hypotheses Rescoring on the CHiME-3 Task. In *Proc. of the IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 409–415, Scottsdale, Arizona, USA.

S. Jalalvand, M. Negri, D. Falavigna, and M. Turchi. 2015b. Driving ROVER With Segment-based ASR Quality Estimation. In *Proc. of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1095–1105, Beijing, China.

Y. Mehdad, M. Negri, and M. Federico. 2012. Match without a Referee: Evaluating MT Adequacy without Reference Translations. In *Proc. of the Machine Translation Workshop (WMT2012)*, pages 171–180, Montréal, Canada.

T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur. 2010. Recurrent Neural Network Based Language Model. In *Proc. of INTERSPEECH*, pages 1045–1048, Makuhari, Chiba, Japan.

M. Negri, M. Turchi, J. G. C. de Souza, and F. Daniele. 2014. Quality Estimation for Automatic Speech Recognition. In *Proc. of the 25th International Conference on Computational Linguistics: Technical Papers (COLING)*, pages 1813–1823, Dublin, Ireland.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. 2011. Scikit-learn: Machine Learning in Python. *Machine Learning Research*, 12:2825–2830.

D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely. 2011. The Kaldi Speech Recognition Toolkit. In *Proc. of the IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, Hawaii, USA.

C. Servan, N.-T. Le, N. Q. Luong, B. Lecouteux, and L. Besacier. 2015. An open source toolkit for word-level confidence estimation in machine translation. In *Proc. of the 12th International Workshop on Spoken Language Translation (IWSLT)*, Vietnam.

K. Shah, M. Turchi, and L. Specia. 2014. An Efficient and User-friendly Tool for Machine Translation Quality Estimation. In *Proc. of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland.

L. Specia, K. Shah, J. G. de Souza, and T. Cohn. 2013. Quest - a translation quality estimation framework. In *Proc. of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 79–84, Sofia, Bulgaria. Association for Computational Linguistics.

A. Stolcke, H. Bratt, J. Butzberger, H. Franco, V. R. Gadde, M. Plauche, C. Richey, E. Shriberg, K. Sonmez, F. Weng, and J. Zheng. 2000. The SRI March 2000 HUBS Conversational Speech Transcription System. In *Proc. of the NIST Speech Transcription Workshop*.

F. Wessel, R. Schluter, K. Macherey, and H. Ney. 2001. Confidence Measures for Large Vocabulary Continuous Speech Recognition. *IEEE Transactions on Audio, Speech and Language Processing*, 9(3):288–298.