Punctuation Insertion for Real-time Spoken Language Translation

Eunah Cho, Jan Niehues, Kevin Kilgour, Alex Waibel

Institute for Anthropomatics and Robotics
Karlsruhe Institute of Technology, Germany
{eunah.cho|kevin.kilgour|jan.niehues|alex.waibel}@kit.edu

Abstract

Sentence segmentation and punctuation insertion in the output of automatic recognition systems is essential for its readability as well as for the performance of subsequent applications, such as machine translation systems. While a longer context can boost the accuracy of inserted punctuation marks, it drastically increases the delay in the spoken language translation system.

In this work, we investigate the impact of shorter context in punctuation insertion task on simultaneous speech translation system. We suggest a new scheme within stream decoding where the time delay consumed on punctuation prediction is avoided. Our evaluations on the English TED talks show that our suggested scheme can be used as an efficient method to punctuate recognized streams in real-time scenarios. While outperforming a conventional language model and prosody based punctuation prediction system, our model maintains a comparable performance compared to systems that require longer contexts.

1. Introduction

Inserting reliable punctuation marks and sentence segmentation into automatically recognized transcripts plays an important role in spoken language translation (SLT) systems. Many of the conventional automatic speech recognition (ASR) systems generate either no or unreliable punctuation marks. Without a proper punctuation insertion component, therefore, the automatically recognized output is hard to read for humans. Also, it affects the performance when the ASR output is used in subsequent applications of natural language processing (NLP), such as machine translation (MT) systems. Missing proper punctuation marks especially degrades the performance of MT systems, since most of them are trained using well-structured texts, such as news corpus, where sentence boundaries are clear and well-formed.

One of the commonly used methods for inserting punctuation marks into the ASR output is the language model (LM) and prosody based scheme as discussed in [1]. It has an advantage that it incorporates acoustic features keeping the process relatively fast. Recently, punctuation insertion models using a monolingual translation system [2, 3] have shown the effectiveness in improving the performance of MT systems when they are applied to the ASR output. A monolingual translation system is an MT system which translates non-punctuated input text into punctuated text. The conventional monolingual translation system suggested in previous work uses overlapping window for input. Since it can provide a very long context, a great performance improvement on the MT for ASR outputs can be achieved using this technique. Overlapping windows, however, make the system difficult to be used in real-time scenarios without long latencies.

One indisputably crucial aspect in inserting punctuation marks for real-time speech translation is the time delay. Longer context is preferred for better prediction performance but it causes more delay.

In this paper, we suggest an efficient punctuation insertion scheme for real-time SLT systems, using the monolingual translation system. Our punctuation insertion and sentence segmentation system is designed to take the output of a stream decoding ASR system. The input to the monolingual translation system is modified so that latency can be decreased while maintaining similar translation performance. We performed experiments both on audio streams as well as manual transcripts, in order to give in-depth analysis on the impact of different length of context in the punctuation insertion scheme.

This paper is organized as follows. In Section 2, a brief overview of past research on punctuation insertion for varied scenarios is given. The task of inserting punctuation marks for real-time translation systems and its related challenges are discussed in Section 3. Section 4 describes how we model the punctuation insertion system for real-time speech translation scenario. The systems we used throughout this work are described in detail in 5. Section 6 shows our experimental setups and results, followed by Section 7 where we conclude our discussion.

2. Related Work

In previous work [4], sentence segmentation for ASR output was modeled based on LM probabilities and prosody. The authors emphasized that choosing a proper segment length for the different MT systems boosts the translation performance. In this work, commas and final periods are not considered separately, but together in order to form segment boundaries. A threshold was used to control the average number of segments per sentence.

Recently MT-driven approaches have emerged as an effective method to insert punctuation marks in ASR output. An approach using a modified phrase table was introduced in [5] as a method to restore commas. Sentence boundaries are generated based on a decision tree on the source side. Applied to three different language pairs, their method significantly improved translation performance.

Among different MT-driven techniques to model punctuation marks for spoken language, the monolingual translation system [3, 2] has shown an outstanding performance in improving machine translation quality in evaluation campaigns [6, 7]. Using this system, a non-punctuated source language is translated monolingually into punctuated source language. In [3], authors made in-depth analysis on three different approaches to restore punctuation marks using an MT system. Among the three systems, they achieved their best performance when using the translation system to translate non-punctuated text into punctuated one. In their work, however, it was assumed that reliable sentence boundaries are already given. Therefore, punctuation marks within each of the given sentence boundaries are restored.

Based on the work in [3], authors in [2] extended the system so that sentence boundaries can also be predicted. In order to model the possibility to insert a final period everywhere given a segment, they randomly cut the training data for the monolingual translation system. Also, the test data was prepared with a shifting window of 10 words.

While the work mentioned above focused on enhancing punctuation accuracy or the machine translation performance when using the punctuated ASR output, the authors in [8] made an extensive study on different segmentation strategies and latency. They inserted segments based on various techniques into ASR output for real-time translation experiments. It was shown that a good performance can be achieved when they use the conjunction-based segmentation strategy along with a comma-based segmentation.

The input segment length and machine translation quality are studied in [9]. In this work, a statistical machine translation (SMT) decoder which processes a continuous input stream was suggested. Using the decoder they achieved improved translation quality at relatively low latencies.

3. Real-time Spoken Language Translation

In order to be useful a real-time spoken language translation system has to, among many other challenges, deal with the problem of latency. The latency of a real-time spoken language translation system is the time between when a word is spoken and when its transcription and translation are displayed to the user [1]. If the latency is more than a few seconds then the whole translation system becomes unusable and frustrating for the user. Each component adds to the latency, due to computation time, communication time and required future context.

Communication time can be kept to a minimum by having a fast connection and low overhead between the individual components. Computation time may be reduced by running the components on fast servers with multiple cores and by parallelizing those parts of the individual components that can be. It may also require sacrificing accuracy by using smaller faster models.

In order to reduce the apparent latency the speech recognition component can be configured to output its current best hypothesis about once a second. The displayed output is then often updated by a newer, possibly better, hypothesis. This type of setup has a much higher user acceptance than the alternative setup where the speech recognition component waits until it has a stable hypothesis before outputting it which can sometimes result in 8 or more words appearing at once.

The MT component is even more dependent on context than the speech recognition component and often has to wait for the whole sentence to be recognized before it can be properly translated. A fast enough MT system can re-translate the sentence each time the ASR system recognizes a new word and change the output displayed to the user. For this to work, however, the MT system requires the ASR output to be segmented into proper sentences.

These design decisions for both the ASR component, the MT component and the real-time spoken language translation system as a whole pose some significant challenges for the punctuation prediction component that converts the stream text output stream of the ASR component into proper sentences required for the MT system. A major side affect of the ASR component constantly updating its current hypothesis is that the punctuation prediction component has to deal with possibly changing inputs. It also has to have a fast computation time because the ASR system is sending updates very frequently. As the MT component requires sentence boundary information as soon as possible in order to function properly the punctuation prediction component has function well with only very little future context.

Although the monolingual translation system [2] shows a good performance in the subsequent application, adopting this system for the real-time speech translation system causes an unacceptable amount of latency due to its long shifting window of 10 words. This component alone would introduce more latency into the whole system than the desired total average latency.

4. Model

In order to decrease the delay in the real-time speech translation system, we use a streaming input scheme instead of the overlapping window. In this section, we describe how the streaming input scheme works.

Our in-house stream decoding ASR system stores its recognition in two separate stacks. In one stack it saves its final 1-best list for words $w = \{w_l, \ldots, w_m\}$. Their following words are stored in another stack $v = \{v_{m+1}, \ldots, v_n\}$, which is not the final recognition yet. Since this stack v is flexible depending on the upcoming context, it is updated

based on the context and whenever it is updated, the changes are shown to users.

In our punctuation insertion setup, we introduce another stack for recognized words before w, in order to consider more context. The history stack h is defined as:

$$h = \{h_{l-c}, \ldots, h_{l-1}\} \tag{1}$$

The context c is chosen as four throughout this work. When there are fewer previous words available in the initial part of the recognition, only upto available context is used.

The newly punctuated string is then obtained by

$$w' = m(h + w) \tag{2}$$

where m denotes the monolingual translation system. Its scheme will be described in detail in Section 5.3. Parts of the generated output is taken as the final string.

$$s = \{w'_{l-c}, \ldots, w'_{m-4}\} \tag{3}$$

At the same time the history stack is updated.

$$h = \{w'_{m-3}, \ldots, w'_{m}\} \tag{4}$$

Thereby we input punctuated text into the monolingual translation system and repunctuate it. Although this leads to a slight mismatch between training and testing data, using this way we can guarantee punctuation can be inserted when the longest context is available.

Table 1 shows how an excerpt from an automatically recognized transcript is punctuated in our monolingual translation system scheme. History stack is marked in blue box.

Input	OK but then after a while
Output	OK. But then, after a while,
Input	then, after a while, I realized this is
Output	then, after a while, I realized this is
Input	I realized this is my life this is six months of
Output	I realized this is my life. This is six months of
Input	is six months of my life and
Output	is six months of my life. And
Input	of my life. And this . . .
Output	of my life. And this . . .

Table 1: History stack and punctuation output

For the non-final ASR recognition stack v, we generate the possible output string $m(h + v)$ and show it to users.

An advantage of this model is that while longer history is utilized, the decision on punctuation insertion on the current window can be made instantly, minimizing the time delay consumed on sentence segmentation. Also, by supporting the stream decoding, users can see the updating recognition as well as its most probable punctuation marks within.

and I said, “OK, it ’s the huge file. OK,
I said, “OK, it ’s the huge file. OK, but
said, “OK, it ’s the huge file. OK, but then
OK. it ’s the huge file. OK, but then, after
it ’s the huge file. OK, but then, after a
’s the huge file. OK, but then, after a while,
the huge file. OK, but then, after a while, I
huge file. OK, but then, after a while, I realised
file. OK, but then, after a while, I realised this
OK , but then, after a while, I realized. this is
but then, after a while, I realized. this is my
then , after a while, I realized. this is my life.
after a while, I realized. this is my life. this
a while, I realized. this is my life. this is
while I realized. this is my life. this is six
I realized. this is my life. this is six months
realized . this is my life. this is six months of
this is my life. this is six months of my
is my life. this is six months of my life
my life. this is six months of my life, and
life . this is six months of my life, and this
this is six months of my life. and this fire
is six months of my life. and this fire. so
six months of my life. and this fire. so, I
months of my life. and this fire. so I was
of my life. and this fire. so I was a
my life. and this fire. so I was a little
life , and this fire. so I was a little bit
and this fire. so I was a little bit skeptical
this fire. so I was a little bit skeptical of

Table 2: Output of monolingual translation system with overlapping window of 10

As a comparison, Table 2 shows the actual output of monolingual translation system with overlapping input, for the same segments shown in Table 1. Since the system is using an overlapping window of 10 words, each encountering word (marked in red box) has to be translated 10 times as well. In this overlapping window system [2], each token is translated ten times and a punctuation mark is inserted depending on how often it occurs after this token. For example, augmenting a punctuation mark after the first encountering word *OK* needs previous ten translations.

From the comparison between Table 1 and Table 2, we can observe that the streaming segmentation system can decrease the latency introduced by using the monolingual translation system with overlapping window. While the suggested streaming segmentation will punctuate the given segments in only 5 times of translation, using the overlapping window requires 30 times of translation.

5. System Description

In this section, we discuss the systems we use throughout this work. The English audio data is decoded using our ASR system. A brief description on our LM and prosody based segmenter, which is used as one of the baseline systems, is also given. Once the punctuation marks are inserted using different segmentation strategies, we translate this test data into German in order to measure the performance of the real-time punctuation insertion system.

5.1. English ASR System

The speech recognition is performed using in-house decoder in an online setup. Using a framesize of 32ms and a frameshift of 10ms the audio stream is converted in a stream of 40 dimensional lMel feature vectors.

The hybrid DNN/HMM acoustic model uses a context dependent quinphone setup with three states per phoneme, a left-to-right HMM topology without skip states. The neural network has in input window of ± 6 frames leading to an input layer size of 520 neurons, this is followed by 4 layers of 2k neurons and a finial classification output layer containing just over 8k neurons.

The neural network is pretrained layerwise using denoising autoencoders with a 20 million mini batches. After pretraining the final layer is added, with the output layer using the softmax activation function. The full DNN is then fine-tuned using the newbob learning rate schedule. All training is performed using Theano [10] on the TED [11] and Quaero data [12].

For the language model training texts from various sources such as webdumps, scraped newspapers and transcripts are used. The 120k vocabulary is selected by building a Witten-Bell smoothed unigram language model using the union of all the text sources vocabulary as the language models' vocabulary (global vocabulary). With the help of the maximum likelihood count estimation method described in [13] we found the best mixture weights for representing the tuning set's vocabulary as a weighted mixture of the sources word counts thereby giving us a ranking of all the words in the global by their relevance to the tuning set.

Using this vocabulary language models are built from each of the sources and interpolated using the SRILM toolkit [14] so as to maximally reduce the perplexity of the tuning set.

5.2. LM and Prosody based Segmentation

The language model and prosody based segmenter employs a 4-gram language model trained on punctuated text. In order to predict punctuation marks a context of four words, two prior and two after the possible punctuation mark, is taken into consideration.

The language model is used to calculate three scores. The first one is the score without an inserted punctuation mark as

$$P(w_{i-1}, w_i, w_{i+1}, w_{i+2}) \tag{5}$$

while the second one is the score with a comma.

$$P(w_{i-1}, w_i, \texttt{@COMMA}, w_{i+1}, w_{i+2}) \tag{6}$$

The last one is calculated by followings.

$$P(w_{i-1}, w_i, \texttt{@STOP}, w_{i+1}, w_{i+2}) \tag{7}$$

A dynamic scaling factor is applied to the punctuation mark scores in order to prevent both very short sentences and very long sentences. In parallel to the language model a prosody component searches for pauses over t_θ seconds and then force terminates any sentences.

5.3. Monolingual Translation System

Monolingual translation system for punctuating English data is trained on the English side of the European Parliament data, News Commentary, TED[1], and the common crawl corpus.

As a preprocessing step, the noisy part of the common crawl data is filtered out using an SVM model as described in [15]. After preprocessing is applied, the normalized training data is resegmented randomly so that punctuation marks can be observed in all possible locations in each line.

For the source side of the training, we removed final period, comma, question mark, and exclamation mark. Double quotation marks are also removed as they are relatively frequent in TED talks. In addition to processing the punctuation marks, we also lowercased every single word on the source side. Since automatically recognized words often miss correct case information, we aim to restore the case information altogether with punctuation marks using this one system. Altogether the training data consists of 10.1 million English words.

The Moses package [16] is used to build the phrase table. We build a 4-gram language model on the entire punctuated target side using the SRILM Toolkit [17]. Word alignment is learned automatically using the GIZA++ Toolkit [18]. A bilingual language model [19] is used along with a 9-gram part-of-speech (POS)-based language model. The POS is learned from TreeTagger [20]. In addition to this POS-based language model, we train a $1,000$ class cluster [21] and use the cluster codes for the additional 9-gram language model. The model weights were optimized on the official test set of IWSLT evaluation campaign in 2012.

5.4. English-German MT System

For evaluating our online punctuation insertion schemes, we translate the testsets with different segmentation and punctuation marks into German. For the translation, we use online

[1] http://www.ted.com

English to German phrase-based translation system. The system is trained on the parallel corpus of Europarl, News commentary, TED, and the noise-filtered common crawl data. For the monolingual data we take the News Shuffle corpus. Detailed statistics on corpus can be found in [7].

We build a 4-gram language model on the German side of TED data which is used as an in-domain language model. In addition to this language model, we used a bilingual language model on all available parallel data as described in [19]. Also, we used a 4-gram language model that is built based on cross entropy with the development data. For the in-domain TED data, we applied the cluster algorithm [21]. Once the TED data is clustered into $1,000$ classes, we build a 9-gram language model and used it as an additional model.

In order to address the word order difference between English and German, we use the POS-based reordering [22] along with the tree-based [23] and lexicalized reordering rules. For optimization of the log-linear combination of models, we use minimum error rate training [24].

For evaluating differently segmented testsets, we use the Levenshtein minimum edit distance algorithm [25] in order to align hypothesis against the reference translation.

5.4.1. Phrase Table Preparation

For online translation systems, it is impossible to generate a perfectly fitting phrase table for each input data. Therefore, we build a phrase table based on the vocabulary in the training data. In order to decrease the size of the model for online scenario, we first filtered out words which occurred in the corpus less than four times. Also, phrases that are longer than 4-grams are filtered out as well.

6. Experiments and Results

In order to measure the impact of different segmentation methods and models on MT, we experiment on the official test set of IWSLT evaluation campaign 2013. The English manual transcript of this test data has 993 sentences, or 17.8K tokens. The audio is 2h 16m long.

The proposed streaming punctuating prediction (StreamingInput) system is compared to both a low latency baseline language model and prosody based punctuation prediction (LM, Prosody) system as well the high latency but highly accurate monolingual translation (Baseline) system using a 10 word moving window. Table 3 presents these systems' translation performance of the test data. The numbers are reported in case-sensitive BLEU [26].

In the first row, we first show the translation performance when using the simple LM and prosody based segmentation, available only for the ASR output. In the baseline system, both ASR output and manual transcript are punctuated using the conventional monolingual translation system, using overlapping windows, as shown in [2]. Therefore, the shift window is applied so that each word is translated for ten times. As it is not for online scenario, the phrase table is also generated upon the knowledge of the each test data.

Punctuation	ASR Output	Manual Transcript
LM, Prosody	9.74	-
Baseline	11.18	19.57
StreamingInput	**11.55**	**19.41**

Table 3: Translation performance of the proposed system compared to a fast LM, prosody based model as well as a high latency, but highly performant monolingual system using an overlapping window

We can see that when we use the suggested punctuation insertion scheme, we achieve 11.55 BLEU points in the ASR test data, beating the conventional LM and prosody based model by 1.8 BLEU points. Even though this system is using relatively shorter context and the less-fitting phrase table than the traditional monolingual translation system, the translation performance is comparable with the baseline monolingual translation system's. Although the translation was slightly worse when using this system for punctuating the manual transcript, we achieve an improvement of 0.4 BLEU in the ASR translation task which is its intended use case.

Due to the small model footprint and the use of an efficient MT decoder the stream-based punctuation prediction setup incurs only minimal computational cost, comparable to the punctuation model based on LM and prosody without having much future context requirements. This fast system also allows for updated punctuation when new data is received. As this component does not add further communication overhead, the total latency of the real-time speech translation system is not negatively impacted.

7. Conclusions

In this paper, we present a new punctuation insertion scheme for real-time spoken language translation system. Taking streamed input from an ASR decoder, the suggested scheme can improve the output of the speech translation without negatively impacting the speech translation system's latency. The experiments show that our low-latency real-time punctuation insertion system can achieve a comparable performance to an offline system requiring a large context window.

As future work, we intend to evaluate the system performance on further language pairs. We would also like to investigate the possible integration of neural network and conditional random field-based punctuation prediction models.

8. Acknowledgements

The project leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement n° 645452.

9. References

[1] E. Cho, C. Fügen, T. Herrmann, K. Kilgour, M. Mediani, C. Mohr, J. Niehues, K. Rottmann, C. Saam, S. Stker, and A. Waibel, "A real-world system for simultaneous translation of german lectures," in *INTERSPEECH*, Lyon, France, 2013.

[2] E. Cho, J. Niehues, and A. Waibel, "Segmentation and Punctuation Prediction in Speech Language Translation using a Monolingual Translation System," in *IWSLT*, Hong Kong, China, 2012.

[3] S. Peitz, M. Freitag, A. Mauser, and H. Ney, "Modeling Punctuation Prediction as Machine Translation," in *IWSLT*, San Francisco, CA, USA, 2011.

[4] S. Rao, I. Lane, and T. Schultz, "Optimizing Sentence Segmentation for Spoken Language Translation," in *Proc. of Interspeech*, Antwerp, Belgium, 2007.

[5] M. Paulik, S. Rao, I. Lane, S. Vogel, and T. Schultz, "Sentence Segmentation and Punctuation Recovery for Spoken Language Translation," in *ICASSP*, Las Vegas, Nevada, USA, April 2008.

[6] T.-L. Ha, T. Herrmann, J. Niehues, M. Mediani, E. Cho, Y. Zhang, I. Slawik, and A. Waibel, "The KIT Translation Systems for IWSLT 2013," in *Proceedings of the International Workshop for Spoken Language Translation (IWSLT 2013)*, Heidelberg, Germany, 2013.

[7] I. Slawik, M. Mediani, J. Niehues, Y. Zhang, E. Cho, T. Herrmann, T.-L. Ha, and A. Waibel, "The KIT Translation Systems for IWSLT 2014," in *Proceedings of the International Workshop for Spoken Language Translation (IWSLT 2014)*, Lake Tahoe, CA, USA, 2014.

[8] V. K. R. Sridhar, J. Chen, S. Bangalore, A. Ljolje, and R. Chengalvarayan, "Segmentation strategies for streaming speech translation," in *HLT-NAACL*, 2013, pp. 230–238.

[9] M. Kolss, S. Vogel, and A. Waibel, "Stream decoding for simultaneous spoken language translation."

[10] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, "Theano: a CPU and GPU math expression compiler," in *Proceedings of the Python for Scientific Computing Conference (SciPy)*, June 2010, oral Presentation.

[11] M. Cettolo, J. Niehues, S. Stüker, L. Bentivogli, and M. Federico, "Report on the 10th iwslt evaluation campaign," in *Proceedings of the International Workshop on Spoken Language Translation, Heidelberg, Germany*, 2013.

[12] S. Stüker, K. Kilgour, and F. Kraft, "Quaero 2010 speech-to-text evaluation systems," in *High Performance Computing in Science and Engineering'11*. Springer, 2012, pp. 607–618.

[13] A. Venkataraman and W. Wang, "Techniques for effective vocabulary selection," *arXiv preprint cs/0306022*, 2003.

[14] A. Stolcke, "Srilm-an extensible language modeling toolkit," in *Seventh International Conference on Spoken Language Processing*, 2002.

[15] M. Mediani, E. Cho, J. Niehues, T. Herrmann, and A. Waibel, "The kit english-french translation systems for iwslt 2011," in *Proceedings of the eight International Workshop on Spoken Language Translation (IWSLT)*, 2011.

[16] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst, "Moses: Open Source Toolkit for Statistical Machine Translation," in *ACL, Demonstration Session*, Prague, Czech Republic, 2007.

[17] A. Stolcke, "SRILM – An Extensible Language Modeling Toolkit." in *ICSLP*, Denver, CO, USA, 2002.

[18] F. J. Och and H. Ney, "A Systematic Comparison of Various Statistical Alignment Models," *Computational Linguistics*, vol. 29, no. 1, pp. 19–51, 2003.

[19] J. Niehues, T. Herrmann, S. Vogel, and A. Waibel, "Wider Context by Using Bilingual Language Models in Machine Translation," in *WMT*, Edinburgh, UK, 2011.

[20] H. Schmid, "Probabilistic Part-of-Speech Tagging Using Decision Trees," in *International Conference on New Methods in Language Processing*, Manchester, UK, 1994.

[21] F. J. Och, "An efficient method for determining bilingual word classes," in *Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics*, ser. EACL '99. Stroudsburg, PA, USA: Association for Computational Linguistics, 1999, pp. 71–76. [Online]. Available: http://dx.doi.org/10.3115/977035.977046

[22] K. Rottmann and S. Vogel, "Word Reordering in Statistical Machine Translation with a POS-Based Distortion Model," in *TMI*, Skövde, Sweden, 2007.

[23] T. Herrmann, J. Niehues, and A. Waibel, "Combining Word Reordering Methods on different Linguistic Abstraction Levels for Statistical Machine Translation," in *Proceedings of the Seventh Workshop on Syntax, Semantics and Structure in Statistical Translation*, Altanta, Georgia, USA, 2013.

[24] A. Venugopal, A. Zollman, and A. Waibel, "Training and Evaluation Error Minimization Rules for Statistical Machine Translation," in *WPT*, Ann Arbor, MI, USA, 2005.

[25] E. Matusov, G. Leusch, O. Bender, and H. Ney, "Evaluating Machine Translation Output with Automatic Sentence Segmentation," in *Proceedings of the International Workshop on Spoken Language Translation (IWSLT)*, Boulder, Colorado, USA, October 2005.

[26] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a Method for Automatic Evaluation of Machine Translation." IBM Research Division, T. J. Watson Research Center, Tech. Rep. RC22176 (W0109-022), 2002.