IMPROVED PUNCTUATION RECOVERY THROUGH COMBINATION OF MULTIPLE SPEECH STREAMS

João Miranda[1,2], *João Paulo Neto*[1] *and Alan W Black*[2]

[1]INESC-ID / Instituto Superior Técnico, Lisboa, Portugal
[2]School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA
jrsm@l2f.inesc-id.pt, Joao.Neto@inesc-id.pt, awb@cs.cmu.edu

ABSTRACT

In this paper, we present a technique to use the information in multiple parallel speech streams, which are approximate translations of each other, in order to improve performance in a punctuation recovery task. We first build a phrase-level alignment of these multiple streams, using phrase tables to link the phrase pairs together. The information so collected is then used to make it more likely that sentence units are equivalent across streams. We applied this technique to a number of simultaneously interpreted speeches of the European Parliament Committees, for the recovery of the full stop, in four different languages (English, Italian, Portuguese and Spanish). We observed an average improvement in SER of 37% when compared to an existing baseline, in Portuguese and English .

Index Terms— speech recognition, machine translation, punctuation, multistream, combination

1. INTRODUCTION

The automatic recovery of punctuation from speech recognition output is of great importance, not only because it enhances the human readability of the produced text, but also because it has a positive impact on a number of spoken language understanding tasks. For instance, parsers or machine translation systems often perform better if provided with well-formed sentences as input [1], and are able to extract useful information from the location of commas and final stops. The detection of final stops is not always easy, since there isn't a one-to-one correspondence between the location of speaker pauses and that of sentence boundaries. The recovery of other punctuation marks can be harder, since they either occur infrequently (e.g. the question mark, the exclamation mark, or the semi-colon) or their use is relatively ambiguous or inconsistent across human annotators.

Traditionally, this has been done by considering a number of features of the input speech. Prosodic features, that look at pause duration, the energy of a given segment, the length of the final word, or pitch information, are among the most effective indicators of the end of a sentence [2]. Another broad class of features are lexical features, which are extracted from the speech recognizer output and capitalize on the fact that some word sequences are more likely to precede or follow a given punctuation symbol than others. One can then train a classifier to combine the information from these features and decide which punctuation mark, if any, should be inserted at a given word boundary. Different ways to do this include using a maximum entropy model [3] or an HMM-based framework where the hidden events are the punctuation marks to be recovered [4].

In a number of cases, however, there might be additional information available, which if used could help improve the automated recovery of punctuation marks. In these application areas, which include meetings of supranational organizations such as the United Nations or the European Union, as well as subtitled TV shows, one has access to multiple redundant streams which are, in a sense, approximate translations of each other. In the case of subtitled TV shows, the two streams are the text extracted from the subtitles and the original speech, whereas when interpretation is available, the streams contain the speech produced by the original speaker together with the translated versions. Since we expect similar information to be present in each of these redundant streams, we also expect a certain degree of equivalence to exist between sentence units among the different streams. To take advantage of this equivalence, however, we first need to combine the information that is contained in the parallel streams. The most straightforward way to do this is to use machine translation techniques to map word sequences in one stream to word sequences in another.

The effective integration of Automatic Speech Recognition (ASR) and Machine Translation (MT) systems has been the focus of many recent studies. In some approaches, the two components are joined sequentially : the output of the ASR system is used as input to the MT system, usually for an application such as speech translation. One of the key ideas in this combination is to allow the MT component, using different language and translation models, to recover from some of the errors of the recognizer, so the ASR system usually keeps track of a number of alternatives to its best hypothesis, in the form of N-best lists, confusion networks [5] or lattices. Parallel integration of ASR and MT models has also been investigated by some authors. Here, the idea is that one or more ASR systems are run in parallel, on streams that are, partially or as a whole, translations of each other. These systems also use the data structures mentioned above to keep track of multiple hypotheses, and as a way of exchanging information between the multiple streams. A significant number of these methods combine a speech stream and a text stream, usually for an application such as machine-aided human translation [6, 7], where a human dictates a translation to a text, and the ASR language model for that task is adapted with the expected translation, generated from a statistical alignment model. A small number of works have also considered combining multiple parallel speech streams [8, 9].

In previous work [10], we developed a general method to combine multiple parallel data streams, containing speech or text, with the idea of improving speech recognition performance for those streams containing speech. For each stream, we generate a lattice which represents a probability distribution over the possible word sequences, and we use phrase tables to match word sequences across pairs of streams. We then use this information to create an alignment of phrase pairs that connects an unrestricted number of streams. We then project the generated alignment into each of the streams that we wish to rescore, and we bias the recognizer's language model towards the resulting set of phrases, therefore improving ASR performance. The alignment generated by this method serves as a starting point for the current work and is discussed in Section 2. Since the lattices cannot contain all the possible word sequences, we also developed methods to recover phrase pairs that would otherwise be lost, and pronunciations differing from those in the existing dictionary [11].

In this paper, we propose to improve performance in a different task, by using the information extracted from an alignment of multiple parallel streams in order to more accurately place full stops in ASR transcripts .

The rest of this paper is organized as follows. Section 2 describes our baseline systems: the ASR and SMT systems used, the multistream combination framework and the punctuation and capitalization system. Section 3 introduces the developed method, which combines the information in multiple streams in order to better recover full stops in each of these streams. In Section 4, we evaluate the improvements of our method. Finally, in Section 5, we draw conclusions and suggest possible future work.

2. BASELINE SYSTEMS

2.1. ASR and SMT systems description

We used four languages - English, Italian, Portuguese and Spanish - to develop our ASR and SMT systems. We used Audimus [12], a hybrid ANN-MLP WFST-based recognizer, as the speech recognizer for this work. We trained 4-gram language models for each of the languages using the Europarl Parallel Corpus [13], and used our existing acoustic models and lexica for these four languages [14]. Phrase tables were created for the six possible language combinations (Portuguese-Spanish, Portuguese-English, Portuguese-Italian, Spanish-English, Spanish-Italian and English-Italian). We used the Moses toolkit [15] to train these phrase tables on the European Parliament Parallel data.

2.2. Multistream Combination Framework

The architecture of our baseline multistream combination framework can be seen in Figure 1. The combination method [10] consists of the following steps:

- Generate phrase tables for each of the stream pairs that we wish to combine. While this can be done by training phrase tables from parallel corpora in the languages of the two streams, depending on the stream types one could also use rule based mappings.
- Use a speech recognizer in order to generate transcriptions, as well as lattices, for the speech streams; specific methods for generating lattices need to be provided for other streams, such as those containing text.
- Compute a posterior probability distribution over all the n-grams, with n less than a fixed threshold, from the generated lattices.
- For every stream pair, compute the intersection between the lattices and the appropriate phrase table, obtaining a set of phrase pairs common to both lattices and the phrase table.
- Rescore the phrase pairs from the previous step, estimating their likelihood of actually having appeared in the speech. The highest-scoring among these pairs are used to construct a phrase pair alignment.
- Rescore the lattices and produce new transcriptions for those streams, such as speech, that can be rescored, using the alignment generated in the previous step.

2.2.1. Intersection between lattices and phrase tables

The intersection step selects the phrase pairs $source \; ||| \; target$ that simultaneously are in the phrase table and for which both $source$ and $target$ can be found in the source and target lattices, respectively. The source and target phrases must be occur sufficiently close in terms of time. The definition of "close" depends on the types of the two streams that are being intersected, although for two speech streams it can usually be a fixed delay parameter. The efficient computation of this intersection uses a specialized algorithm [10].

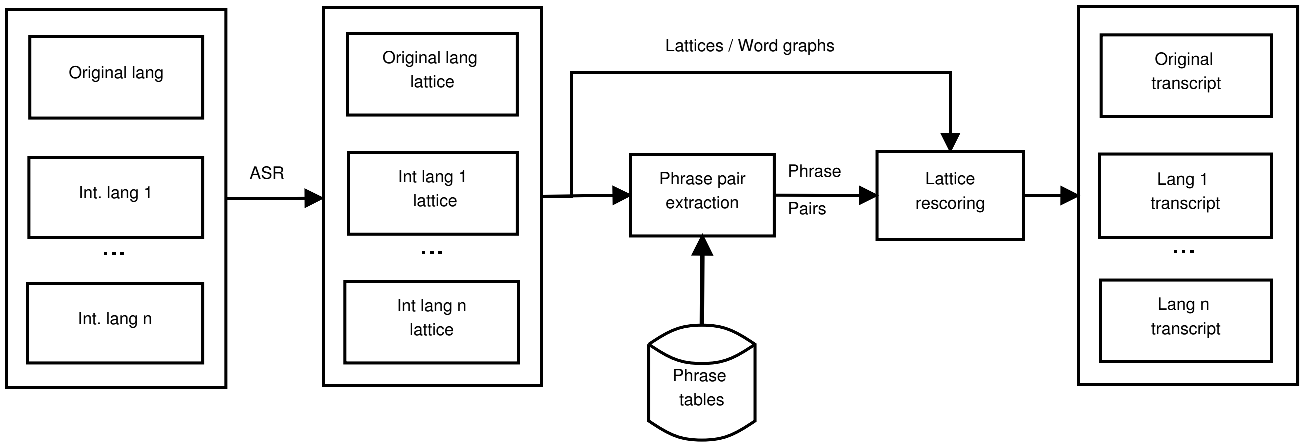

Fig. 1. The proposed system architecture [10]

🇵🇹	há	10.4-10.5	-0.0	🇬🇧	there are	11.3-12.3	-6.5
🇵🇹	há várias	10.4-10.8	-0.1	🇪🇸	hay muchas	10.2-10.7	-0.2
🇵🇹	opções	10.8-11.2	-0.0	🇬🇧	options	12.6-13.9	-8.8
🇪🇸	opciones	10.7-11.2	-0.0	🇬🇧	options	12.6-13.9	-8.8

Fig. 2. Sample of a non-conflicting alignment of three streams, shown as a list of phrase pairs. For each phrase in the list, the time at which it occurs in the stream (e.g. 10.4-10.5), and its posterior probability (e.g. -0.0), generated from the ASR lattices, are also displayed.

2.2.2. Alignment generation

Because most of the phrase pairs that are extracted do not actually occur in the respective streams, we perform a preliminary filtering step, in order to eliminate these undesirable phrase pairs. We classify each phrase pair on whether or not it should be removed, based on a number of features, such as the posterior probabilities of each of the phrases in the phrase pair, its phrase table features or the language model scores of the words in each phrase. We then generate an alignment between the streams by selecting the highest scoring, non-overlapping phrase pairs from the filtered list. A sample of such an alignment can be seen in Figure 2.

2.2.3. Lattice rescoring

In order to obtain new transcriptions for streams containing speech, the generated alignment is first projected into the stream that we want to rescore. For instance, if we wanted to rescore the English stream in Figure 2, we would get the following phrases: 'options (12.6s-13.9s)' and 'there are (11.3s-12.3s)'. Then a rescoring of the lattices for each stream is carried out, in order to generate new transcripts, where the language model of the speech recognizer is biased towards the phrases in the generated projection.

2.3. Punctuation System

The baseline punctuation system [16] consists of a maximum entropy classifier, which combines a number of features that are extracted from the region surrounding each potential sentence boundary (which corresponds to each word boundary).

These include word features, which capture information about which words or word bigrams are likely to occur close to a sentence boundary; speaker identity features, which use the information provided by a speaker clustering system to detect if the speaker has changed; POS features, which consider the tags assigned to words by a part-of-speech tagger; the segments produced by an acoustic segmenter, and the duration of the intervals between consecutive neighbouring words.

When recovering punctuation, most of these features are obtained from the automatically generated speech transcripts or from the output of the audio pre-processor module. The Portuguese and English punctuation systems were trained using manually annotated broadcast news data.

3. PROPOSED METHOD

The main insight behind the proposed method is that we can find an approximate correspondence between sentences in different streams. That is, we expect to find, for a sentence in one stream, a sentence containing equivalent information in each of the other streams. Of course, this may not always be true, but we expect it to be a reasonable approximation. For example, in the case of simultaneous translation of speech, the interpreter may drop a part of a sentence due to being unable to keep up with the speaker, or they may split a sentence into several different sentences. In the latter case, there would no longer be a one-to-one equivalence between sentences in the two streams, so we assume this to be a relatively rare occurrence.

In light of this assumption, our method consists of minimizing the function f in Equation 1, where the binary vector

$$f(s_1 \dots s_n) = \alpha_1 \delta(s_1 \dots s_n) + \alpha_2 \tau(s_1 \dots s_n) - \alpha_3 \gamma(s_1 \dots s_n) - \alpha_4 \left(\sum_{w_{ij}=1} p_{ij} + \sum_{w_{ij}=0} (1 - p_{ij}) \right) \quad (1)$$

$s_i = w_{i1} \dots w_{im}$ represents the segmentation for stream i, such that $w_{ij} = 1$ if there is a full stop before the j^{th} word of stream i, and $w_{ij} = 0$ otherwise, and p_{ij}, when available, is the probability that there is a full stop before the j^{th} word of stream i, given by the baseline classifier. The function f depends on all the segmentations; therefore, by optimizing it we simultaneously improve the segmentations for all the streams.

The function $\delta(s_1 \dots s_n)$ tries to incorporate the information that we obtained from the multistream alignment. It does so by computing the number of distinct conflicts in the sentence segmentation $s_1 \dots s_n$. Consider two phrase pairs, pp_1 and pp_2, which are in the streams s and t. Then pp_1 and pp_2 are said to be in conflict, if there is a sentence boundary between the corresponding phrases in one of the streams but not in the other. For example, suppose we have the two phrase pairs $pp_1 =$ "eles tinham || they had" and $pp_2 =$ "tentado || tried" , in the English and Portuguese streams, and that, in the current segmentation, pp_1 and pp_2 are in different sentences in one of the streams but not in the other. Then the function $\delta(s_1 \dots s_n)$ penalizes this fact, trying to bring the phrase pairs to equivalent sentences in the two streams.

Also, the component $\tau(s_1 \dots s_n)$ assigns a penalty which is proportional to the duration of an interword pause (the difference between the start time of the current word and the end time of the previous word) to word boundaries that are not preceded by a full stop in the current segmentation. The component $\gamma(s_1 \dots s_n)$ represents, for each stream, the n-gram score variation introduced by adding full stops at the locations defined by segmentation $s_1 \dots s_n$. For example, if the original sentence for stream i is "Thank you chair we will now proceed." and the suggested punctuation is "Thank you chair. We will now proceed.", then the contribution of stream i to $\gamma(s_1 \dots s_n)$ is the difference between the LM scores of the second and the first punctuations. Finally, the last component assigns a higher score to punctuations that agree with the baseline classifier (when its information is available).

To optimize function f, we start with some initial joint sentence segmentation $s_1 \dots s_n$, and perform an applicable local operation - a valid operation which improves the total punctuation score given by function f - in order to produce a new sentence segmentation. We then iteratively apply one of these operations until reaching a local minimum (none of the operations can be applied) or a predefined maximum number of iterations. We therefore jointly optimize the punctuations of each of the streams, in a hill-climbing manner. The two types of applicable operations are the following:

- Merging two or more consecutive sentences in the same stream into one, therefore implicitly removing the full stops between them.
- Splitting a sentence, in a given stream, into two sentences, where the possible splitting locations are the word boundaries inside that sentence. This operation corresponds to inserting a full stop in the corresponding punctuation.

Lang	Duration	Num. words
PT	1h23m	8467
ES	1h23m	9097
EN	1h23m	10991
IT	1h23m	8455

Table 1. Duration and number of words of each of the languages in the test set.

All the instantiations of the above operations are sorted by decreasing order of Δf. If more than one operation decreasing f is available, then the one which decreases it the most is selected at each step.

The parameters $\alpha_i > 0$ of Equation 1 are selected to minimize Slot Error Rate (SER), averaged over the resulting segmentations $s_1^* \dots s_n^*$, in a held-out development set.

4. EXPERIMENTAL EVALUATION

4.1. Experimental Setup

We collected and manually transcribed four speeches, in English, from the DEVE, ENVI, LEGAL and IMCO Commitees of the European Parliament, as well as their respective interpreted versions in three other languages (Italian, Portuguese and Spanish). Our development set consists of two other speeches, from the European Parliament Comittees. In Table 1 we see that the interpreted versions have less words than the original one, indicating the extent to which the interpreters summarized the original speech.

We automatically transcribed and generated ASR lattices for each of the speeches, and then executed the multistream combination algorithm of Section 2.2, in order to generate an alignment of the four streams. This alignment was subsequently used as input to the proposed method.

4.2. Results

To evaluate the impact of our proposed method on punctuation recovery performance, we computed four different performance metrics : Slot Error Rate (SER) [17], which represents the number of insertions, deletions and substitutions of punctuation marks divided by the total number of reference punctuation marks, precision (P), recall (R) and F-measure

	SNS				Baseline				Prop. method			
	P	R	F1	SER	P	R	F1	SER	P	R	F1	SER
PT	0.259	0.817	0.393	2.540	0.372	0.365	0.365	1.281	0.632	0.604	0.616	0.761
ES	0.209	0.762	0.323	3.259	-	-	-	-	0.618	0.575	0.587	0.819
EN	0.338	0.946	0.489	2.205	0.558	0.693	0.614	0.897	0.710	0.718	0.710	0.608
IT	0.323	0.828	0.455	2.172	-	-	-	-	0.557	0.609	0.574	0.943
All	0.283	0.838	0.415	2.544	0.465	0.529	0.490	1.089	0.629	0.626	0.622	0.783

Table 2. The different performance metrics that were evaluated, for each of the tested methods, across Portuguese (PT), Spanish (ES), English (EN) and Italian (IT), and averaged over the test set. The table entries corresponding to Spanish and Italian are not available, since we lacked training data to generate instances of the classifier for these two languages. Therefore, the values in the "All" row are not directly comparable between the Baseline method and the other techniques.

(F1), the harmonic mean of precision and recall. To do this, we first align the automatic transcripts with the manual reference. The values of these metrics for the proposed method were compared with our two baselines. The first baseline was our speech / non-speech component (SNS) [18], which splits speech into segments, based mostly on speech activity; we considered these segments to be implicitly delimited by full stops. The second baseline was the punctuation and capitalization system described in Section 2.3; we add a full stop to the output wherever the probability generated by this system is greater than 0.5. This baseline was only available for the Portuguese and English languages. The output of the SNS component was also used to initialize the search algorithm described in Section 3. All the comparisons were carried out for each of the four languages considered.

The main results are summarized in Table 2. Overall, the method that produced the poorest results was - unsurprisingly, since it was not designed for the purpose of punctuation - the SNS component. It had the highest average recall among all the three tested methods together with the highest SER, which suggests that it split speech into a very large number of sentences, creating many spurious insertions of full stops. Also, the proposed method has the best results across all of the languages that we considered. Compared with the baseline method in Portuguese, there is a 40% reduction in SER and a 68% improvement in F1, and in English there is a 32% reduction in SER and a 15% improvement in F1.

We observe that all of the three compared methods are more effective in the original speech (always given in English), across all the considered metrics, than in the interpreted versions. Not only do the interpreters often pause at locations that are unrelated to the sentence boundaries, they sometimes also produce larger numbers of disfluencies which may be confusing as to the location of sentence boundaries. Also, the spontaneous speech they produce is usually recognized at a higher word error rate, and this disrupts features that are based on word identities, such as language model scores.

It is also interesting to note that the proposed method performs slightly worse in the interpreted languages (Spanish and Italian) for which the probabilities from the baseline classifier are unavailable, when compared to Portuguese, which suggests that including these as features had a positive impact on the proposed method. In fact, by inspecting Table 3, we find that this is actually the case: the use of these probabilities improves SER about 2.4% absolute and, while the largest improvements are for English, all the languages show improvements, even those (Spanish and Italian) for which the probabilities of the baseline classifier were not available.

	SER (no baseline prob.)	SER (with baseline prob.)
PT	0.780	0.761
ES	0.832	0.819
EN	0.652	0.608
IT	0.966	0.943
All	0.807	0.783

Table 3. Comparison between the average SER of the different languages, depending on whether the baseline classifier probabilities, for PT and EN, are used as features.

5. CONCLUSIONS

We presented a technique for integrating the information in multiple parallel streams, in order to improve the performance of automatic punctuation recovery from ASR transcripts. Our method extracts the information contained in an alignment that joins phrases in the various streams, and uses it to guide the better placement of end-of-sentence marks. It does not require any training data, apart from a small development set used to tune a number of parameters of the algorithm. We evaluated our method in a test set consisting of European Parliament Committee speeches, in four languages (English, Italian, Spanish and Portuguese). We obtained an average 37% improvement in SER, when compared with a maximum entropy baseline, considering the two languages (Portuguese and English) for which this baseline was available.

In the future, we would like to extend this work to incorporate the recovery of different punctuation marks, such as the comma, question mark or exclamation mark. For example, recovery of the question mark, which is easier in languages in which interrogatives are identified by cues such as subject-verb inversion, could be made more effective by sharing this information across languages. We would also like to focus on a different topic in rich transcription of speech - improved automatic detection and recovery of disfluencies such as filled pauses, repetitions or edits, a task which would also benefit from the combination of multiple parallel information streams. We expect that detecting and recovering these speech artifacts will contribute to the task of better recovering punctuation, as the two problems are mutually interlinked.

6. ACKNOWLEDGEMENTS

Support for this research was provided by the Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) through the Carnegie Mellon Portugal Program under Grant SFRH/BD/33767/2009, and through project PEst-OE/EEI/LA0021/2013. The work leading to these results has received funding from the European Community under grant agreement number 296371.

7. REFERENCES

[1] Fredrik Jørgensen, "The Effects of Disfluency Detection in Parsing Spoken Language," in *Proceedings of the 16th Nordic Conference of Computational Linguistics NODALIDA-2007*, Tartu, Estonia, 2007.

[2] J. G. Fung, D. Z. Hakkani-Tür, M. Magimai-Doss, E. Shriberg, S. Cuendet, and N. Mirghafori, "Cross-Linguistic Analysis of Prosodic Features for Sentence Segmentation," in *Proceedings of the Interspeech*, 2007.

[3] J. Huang and G. Zweig, "Maximum entropy model for punctuation annotation from speech," in *Proceedings of ICSLP*, Denver, Colorado, USA, 2002.

[4] H. Christensen, Y. Gotoh, and S. Renals, "Punctuation annotation using statistical prosody models," in *Proceedings of the ISCA Workshop on Prosody in Speech Recognition and Understanding*, 2001.

[5] L. Mangu, E. Brill, and A. Stolcke, "Finding consensus in speech recognition: word error minimization and other applications of confusion networks," *Computer Speech and Language*, vol. 14, no. 4, pp. 373–400, 2000.

[6] S. Khadivi and H. Ney, "Integration of Speech Recognition and Machine Translation in Computer-Assisted Translation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 8, pp. 1551–1564, 2008.

[7] A. Reddy and R.C Rose, "Integration of statistical models for dictation of document translations in a machine-aided human translation task," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 8, pp. 2015–2027, 2010.

[8] M. Paulik, S. Stüker, C. Fügen, T. Schultz, T. Schaaf, and A. Waibel, "Speech Translation Enhanced Automatic Speech Recognition," in *Proceedings of the ASRU*, San Juan, Puerto Rico, 2005.

[9] M. Paulik and A. Waibel, "Extracting Clues from Human Interpreter Speech for Spoken Language Translation," in *Proceedings of ICASSP*, Las Vegas, USA, 2008.

[10] J. Miranda, J. P. Neto, and A. W Black, "Parallel combination of speech streams for improved ASR," in *Proceedings of the Interspeech*, Portland, USA, 2012.

[11] J. Miranda, J. P. Neto, and A. W Black, "Recovery of acronyms, out-of-lattice words and pronunciations from parallel multilingual speech," in *Proceedings of the SLT*, Miami, USA, 2012.

[12] H. Meinedo, D. A. Caseiro, J. P. Neto, and I. Trancoso, "AUDIMUS.media: a Broadcast News speech recognition system for the European Portuguese language," in *Proceedings of PROPOR*, Faro, Portugal, 2003.

[13] P. Koehn, "Europarl: A Parallel Corpus for Statistical Machine Translation," in *Proceedings of the tenth Machine Translation Summit*, Phuket, Thailand, 2005.

[14] H. Meinedo, A. Abad, T. Pellegrini, I. Trancoso, and J. P. Neto, "The L2F Broadcast News Speech Recognition System," in *Proceedings of Fala2010*, Vigo, Spain, 2010.

[15] 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 *Proceedings of the ACL demo session*, Prague, Czech Republic, 2007.

[16] F. Batista, H. Moniz, I. Trancoso, and N. J. Mamede, "Bilingual Experiments on Automatic Recovery of Capitalization and Punctuation of Automatic Speech Transcripts," *IEEE Transactions on Audio, Speech and Language Processing*, pp. 474–485, 2012.

[17] J. Makhoul, F. Kubala, R. Schwartz, and R. Weischedel, "Performance measures for information extraction," in *Proceedings of the DARPA Broadcast News Workshop*, 1999, pp. 249–252.

[18] H. Meinedo and J. P. Neto, "Audio segmentation, classification and clustering in a broadcast news task," in *Proceedings of ICASSP*, Hong Kong, China, 2003.