Computer-assisted transcription of speech based on confusion network reordering

Antoine Laurent[†§]*, Sylvain Meignier*[†]*, Teva Merlin*[†]*, Paul Deléglise*[†]

[†]LIUM (Computer Science Research Center – Université du Maine) – Le Mans, France
[§]Spécinov – Trélazé, France
first.last@lium.univ-lemans.fr, a.laurent@specinov.fr

Abstract

Large vocabulary automatic speech recognition (ASR) technologies perform well in known and controlled contexts. In less controlled conditions, however, human review is often necessary to check and correct the results of such systems in order to ensure that the output of ASR will be understandable. We propose a method for computer-assisted transcription of speech, based on automatic reordering confusion networks. Our method will be evaluated in terms of KSR (Keystroke Saving Rate) and WSR (Word Stroke Ratio). It allows to significantly reduce the number of actions needed to correct ASR outputs. WSR computed before and after every network reordering shows a gain of about 17.7% (3.4 points).

Index Terms: Speech recognition, Automatic correction, Cache models, Confusion network

1. Introduction

Automatic Speech Recognition (ASR) systems are used to get a textual transcription of a speech signal. Adverse conditions, such as a noisy environment, spontaneous speech, etc., result in a high number of transcription errors that often render human post-processing highly desirable, if not mandatory. A human transcriptor is then in charge of reading and correcting the ASR output. Experiments carried out by [1] show that this approach allows to significantly reduce the time needed to get an error-free document, comparatively to manually typing the whole document.

The method proposed here aims to dynamically help human correctors of automatically generated transcriptions. It relies on collaboration between the corrector and the ASR system. Every time the user corrects a word in the automatic transcription, this correction is immediately taken into account to re-evaluate the transcription of the words following it. As a result of the re-evaluation, the system may propose a new transcription for these words that would be more consistent with the newly corrected word, thus potentially saving the corrector from having to correct these words.

This kind of approach has been proposed in the literature [2] under the name Computer-Assisted Transcription of Speech (CATS) but has not been the focus of many articles yet. Similar techniques have been used in the domain of computer-assisted translation: in [3, 4, 5], when the user corrects a word of the automatic translation proposed to him, the system comes up with an alternative translation. Other works [6, 7, 8] present computer-assisted translation systems that use speech utterances as input. The general idea consists in using a combination of an n-gram language model and translation probabilities of the words.

For CATS, the authors of [2] described a method that consists in re-launching the speech recognition process after each user correction. The obvious drawback of this technique is its time consumption, which would have a negative impact on reaction times in an interactive context.

Our intent in this paper is to propose a method for CATS which will run fast enough to be usable in an interactive correction application, providing immediate feedback to the user after each manual correction.

2. Method

The proposed method consists in the re-evaluation of the best hypothesis contained in a confusion network, according to the corrections made by the user. No new signal decoding is necessary.

2.1. Confusion networks

A confusion network is obtained from word lattices constructed during the decoding step. Word lattices represent (after pruning) every developed hypothesis path, in which every state corresponds to a time in the recording to transcribe, and every arc represents a word, associated with a probability.

The transformation of a word lattice into a confusion network is illustrated in Figure 1. It consists in merging locally identical words, grouping temporally close words into confusion sets, and removing weak hypothesis paths [9]. Each word gets a new score equal to its *a posteriori* probability (obtained from the word lattice) divided by the sum of the *a posteriori* probabilities of words. These scores will be used to determine which sequence of words should be provided to the human corrector.

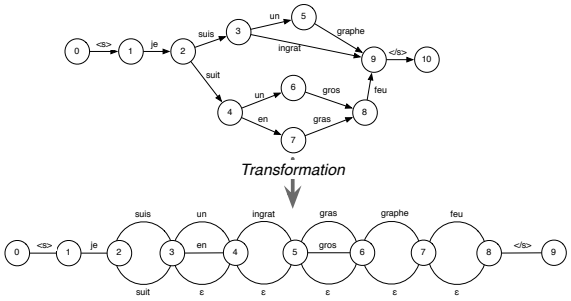

Figure 1: Transforming a word lattice into a confusion network (word probabilities are not shown). The symbol ϵ represents an empty transition (absence of word).

In this work, confusion networks are extended with approximate start and end times for each state, corresponding to the start and end times of the sequence of words associated with the state.

2.2. Principle

The goal of ASR is to determine the most probable word sequence $\hat{W}$ in the set W of all possible word sequences corresponding to the sequence X of acoustic observations. The search of $\hat{W}$ that maximizes the emission probability of W knowing X corresponds to the following equation, after application of the Bayes theorem and simplification:

$$\hat{W} = \arg\max_{W} P(X|W)P(W) \tag{1}$$

$P(W)$ is given by the language model; $P(X|W)$ corresponds to the probability attributed by the acoustic model. In our case, the set W of all possible word sequences is divided in two: a prefix p that was corrected or validated by the user, and a suffix s that we have to find using p. We search the word sequence $\hat{s}$, in the set of all possible suffixes s, that maximizes the following equation:

$$\hat{s} = \arg\max_{s} P(X|s,p)P(s|p) \tag{2}$$

The proposed method does not call the acoustic score into question: $P(X|s,p)$ is constant.

$P(s|p)$ is obtained using a linear combination of a 4-gram language model P_{4G} and a cache model P_{cache} [10]. We search, in the set of all the candidate suffixes $\tilde{s}$ in the confusion network, the suffix $\hat{s}$ that maximizes the following probability:

$$\hat{s} = \arg\max_{\tilde{s}}((1-\lambda)P_{4G}(s|p) + \lambda P_{cache}(s|p)) \tag{3}$$

The cache model is built using words contained in the prefix p. It allows to reinforce the probabilities of recently seen words—we make the hypothesis that these words have a higher probability of appearing in the near future (in s).

In the literature, several cache models are proposed. In our case, the best results in term of perplexity have been obtained by using the method proposed in [10]. The appearance probability of word w_i is exponentially proportional to the distance between the current position and previous positions of word w_i in the history h_i:

$$P_{cache}(w_i|h_i) = \beta \sum_{j=1}^{i-1} I_{\{w_i=w_j\}} e^{-\alpha(i-j)} \tag{4}$$

with α the cost of the gap in the cache model, $I_{w_i=w_j} = 1$ if $w_i = w_j$ (0 otherwise), and with β a normalization constant computed as follows:

$$\beta = \frac{1}{\sum_{j=1}^{i-1} e^{-\alpha j}} \tag{5}$$

We decided to use confusion networks instead of word lattices. Indeed, using lattices would have been complicated in the case when a correction would create a new path not present in the word lattice. Moreover, the confusion networks used have been shrunk in order to avoid combinatorial explosion during the search for candidate suffixes: in our training corpus, there are only 2.12 words per transition on average.

2.3. Application

In the proposed method, the search of candidate suffixes in the confusion network is triggered when—and only when—the user substitutes a word with another. If the user decides to delete a word (in the case of an incorrect word inserted between two correct words) or to insert a word (when a word was missing between two correct words), we assume that the next word is correct.

After the user corrects a word, the first step consists in finding the state corresponding to the new word within the confusion network. If this word is present in the network in the place where the correction was made, it will be bound to the corresponding state. Otherwise, the new word will be added to the state of the substituted word.

Then starts a search for every word sequence that can follow this state, taking into account temporal indices. With the end time of the new word is denoted as t, the set of following states in the confusion network is searched for states with a start time higher than or equal to t. The search is recursive: for every concurrent state, we search again every state that can follow it. The use of the start and end times allows to avoid selection of overlapping words.

Equation (3) allows to decide between the various possible sequences. If two sequences have the same probability (which is an exceptional case), the *a posteriori* mean probability of the word sequences is used as the discriminant element.

Automatic reordering stops when the last word in the proposed sequence corresponds to a word that was already in the previous hypothesis.

2.4. Example

Let's consider a case where, for the sentence "Je suis un graphe", the best hypothesis generated by the ASR is "Je suis ingrat feu", with the confusion network shown in Figure 2 below.

The user first replaces the word "ingrat" with the word "un". With respect to the temporal indices present in the confusion network, the possible word sequences are: "je suis un gras feu", "je suis un gros feu" and "je suis un graphe".

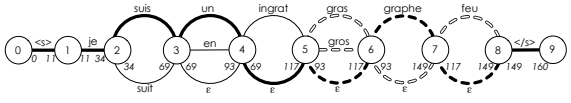

Figure 2: Confusion network: selection of the various possible word sequences. The thick, solid lines represent the error-free prefix after correction by the user. The dashed lines show the possible suffixes.

To choose the new word sequence, we use the cache model interpolated with the language model (as in equation 3). In our example, the probabilities of the sequences "je suis un gros", "je suis un gras" and "je suis un graphe" will be computed and compared. If the sequence "je suis un graphe" gets the highest probability of the three (and thus is proposed to the user), only one user action will have been needed in order to correct the entire sentence (replacing the word "ingrat" with the word "un").

2.5. Out-of-vocabulary words

The list of words that can be automatically proposed is limited to the words present in the confusion network. If the user types a word not present in the 4-gram model, the search in the set of

suffixes will be done by using the probability of the unknown word. This word will be added to the cache model, but not to the confusion network. It will not be possible for this new word to automatically reappear during the rest of the correction process.

3. Experiments

Experiments were carried out to simulate human behavior when correcting the best hypothesis of the confusion network generated by the ASR.

3.1. Corpus & ASR system

The optimization of the cache model coefficients has been realized on the ESTER 1 test corpus. Experiments have been carried out on the ESTER 2 test corpus [11]. These corpora are composed of data recorded from francophone radio stations, with the addition of articles from the French newspaper "Le Monde".

Confusion networks are created from the hypothesis word-graph generated by the LIUM ASR that was developed for the ESTER 2 evaluation campaign. The decoding process takes place in 5 passes (see [12] for more information about it). Without any particular treatment applied to the segments coming from African radio stations, the word error rate for the system on the ESTER 2 test corpus is of 19.2%.

3.2. Metric

The proposed method has been evaluated using two metrics: the Word Stroke Ratio (WSR) [13, 14, 2] and the Keystroke Saving Rate (KSR) [15].

The WSR is a commonly used metric for computer assisted translation methods. The WSR is the number of words that have to be corrected divided by the total number of words in the reference. This rate will be calculated by counting the total number of wrong words to correct. With no correction assistance, the WSR is identical to the WER, if we consider that every incorrect word has to be corrected.

The KSR has been developed for AAC (Augmentative and Alternative Communication) systems, for use by handicapped persons. It is computed as follows:

$$KSR = (1 - \frac{k_p}{k_a}) \times 100 \quad (6)$$

Where k_p is the number of strokes made by the user to write a message, and k_a is the number of strokes that would have been necessary without any help for word composition. Those strokes can be done with a keyboard, or with a particular device that takes into account the user handicap: joystick, blink, etc.

In our case, k_p is the number of actions made by the user to correct the ASR best hypothesis, using a keyboard, and k_a is the number of actions that would have been necessary starting from scratch (without automatically generated hypothesis).

To compute the KSR, we assume that the user will always choose the strategy that allows to correct the transcription in a minimum of actions.

Two strategies are retained to minimize the number of actions: every word of the incorrect area is deleted and replaced by a correct word, or every character that is correct is kept and wrong characters are corrected.

An alignment between the hypothesis generated by the ASR and the reference is done. This alignment is realized at the word level and at the letter level in the incorrect areas.

The cost of each action is as follows:

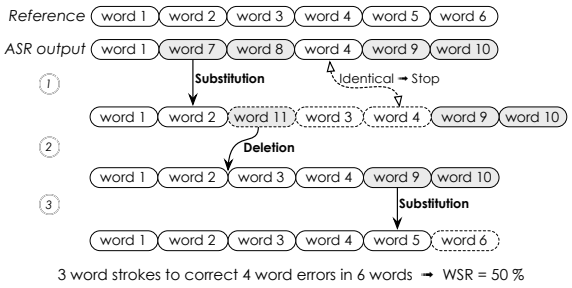

Figure 3: WSR example. User actions are indicated by black arrows. Words with a grey background are errors in the automatic transcription. Word proposed by the reordering method are shown with a dashed outline.

- Moving inside the text from word to word is not taken into account. Computer-assisted translation applications, such as the one proposed in [16], show words one by one during the correction process, and each word is either corrected or validated when it appears.
- Deleting a word counts as 1 action (a shortcut key allows to delete a whole word).
- Hitting a key on the keyboard counts as 1 action.

KSR does not take into account the cost of every action for the user. The proposed method have been evaluated in term of WSR too. The number of incorrect words is counted in every segment until finding a substitution error. This error is counted, and the automatic reordering method triggers and proposes a new hypothesis. The same calculation is repeated until reaching an error-free hypothesis (identical to the reference).

Figure 3 illustrates the computation of WSR. This example shows a case where the reordering method saves the user one correction: the initial ASR output contains 4 wrong words, and only 3 words have to be corrected to get a correct sentence. Hence in this example the WER is $4/7 = 66.7\%$ and the WSR is $3/6 = 50\%$.

3.3. Evaluation method

First, we calculated the number of actions corresponding to manual transcription, and the number of actions for non-assisted correction of the ASR output. Then we evaluated the number of actions using the proposed method, with and without a cache model.

For non-assisted as well as for assisted correction, we also evaluated the impact of a user interface offering the possibility to replace a word by directly selecting another one from a list. The list was composed of words present in the confusion network in the corresponding position. Assuming that this list is always visible on the screen, a cost of 1 is attributed to word substitution with this interface.

[16] shows that this kind of user interface gives good results when used to correct ASR outputs in the case of transcribing TV subtitles. Words appear first in the upper left corner. After a delay, the next word appears to the right of the last word, or on the next line if end of line was reached. This allows the transcriber to focus on one word: once a word is displayed, it never moves. When the lower right corner is reached, the next word replaces the first one in the upper left corner. Words in green (words 5 and 6) are editable, the one in red are not. The editable window moves at the same speed as words appear.

3.4. Results

Tables 1 and 2 show the results obtained in the various configurations.

The reference is composed of 435,005 characters (spaces included). Manual transcription would therefore require 435,005 actions.

For manual correction of ASR outputs with a word error rate of 19.2%, the number of actions decreases to 53,492, yielding a gain in terms of KSR of 87.7%. With the use of the list of words, the number of actions goes down to 52,003, corresponding to a KSR of 88%. In both cases, the WSR is of course equal to the word error rate (19.2%).

Table 1: *KSR et WER without automatic method on ESTER 2 test corpus*

Method	Actions	KSR	WSR
Manual transcription	435,005	0%	–
Manual correction of ASR	53,492	87.7%	19.2%
Manual correction + list	52,003	88.0%	19.2%

Assisted correction of ASR outputs using the automatic reordering method with no cache model allows a KSR of 89.2% (46,795 actions), for a WSR of 17%. Adding the possibility to select words from a list decreases this number to 44,732, for a KSR of 89.7%. Finally, the complete system, using the automatic reordering method with the cache model and the selection of words from a list, yields a KSR of 90.3% (41,992 actions) and a WSR of 15.8%.

Table 2: *KSR and WSR on ESTER 2 test corpus with the proposed method*

Correction Method	Actions	KSR	WSR
Reordering (no cache)	46,795	89.2%	17.0%
Reordering (no cache) + list	44,732	89.7%	17.0%
Reordering with cache + list	41,992	90.3%	15.8%

In terms of WSR, the proposed method allows a gain of about 17.7% (3.4 points) on the ESTER 2 test corpus over what is achieved through non-assisted correction (19.2%).

4. Conclusion

This paper presents an automatic reordering of ASR hypotheses. This method allows a gain of 3.4 points in WSR over correction with no automatic reordering, and allows to decrease the number of actions necessary to correct ASR outputs by 21.5%.

It will be interesting to evaluate this method by developing an application based on the interface proposed here. This will allow to measure time gained through the use of the automatic reordering technique.

In the future, the method could be improved by allowing to add new words to the confusion networks that are used to reorder the hypotheses. Words would be added to states of the networks corresponding to words that are phonetically close.

Another direction worth exploring is propagation of corrections made by users. At this time, correction of one word triggers automatic reordering of confusion network hypotheses only for the end of the segment being corrected. The correction could have an impact on others segments further down in the transcription. A more global approach would aim to reuse previously corrected transcriptions to learn new models and reduce the number of errors of the ASR system, or to automatically apply corrections that were frequently repeated by the user.

5. References

[1] T. Bazillon, V. Jousse, F. Béchet, Y. Estève, G. Linarès, and D. Luzzati, "La parole spontanée : transcription et traitement," in *TAL*, vol. 49, 2008, pp. 47–76.

[2] L. Rodríguez, F. Casacuberta, and E. Vidal, "Computer Assisted Transcription of Speech," in *IPRIA*, vol. 4477, 2007, pp. 241–248.

[3] J. Civera, J. Vilar, E. Cubel, A. Lagarda, S. Barrachina, F. Casacuberta, and E. Vidal, "A novel approach to computer assisted translation based on finite-state transducters," in *FSMNLP*, vol. 4002, 2005, pp. 32–42.

[4] J. Tomás and F. Casacuberta, "Statistical phrase-based models for interactive computer-assisted translation," in *ACL*, 2006, pp. 835–841.

[5] G. Foster, "Text prediction for translators," Ph.D. dissertation, Université de Montréal, Canada., 2002.

[6] J. C. Amengual, J. M. Benedí, A. Castaño, A. Castellanos, V. M. Jiménez, D. Llorens, A. Marzal, M. Pastor, F. Prat, E. Vidal, and J. Vilar, "The EuTrans-I speech translation system," in *Machine Translation*, vol. 14, 2000, pp. 941–951.

[7] E. Vidal, F. Casacuberta, L. Rodríduez, J. Civera, and C. Martnez, "Computer-assisted translation using speech recognition," in *TASLP*, vol. 14, 2006, pp. 941–951.

[8] F. Casacuberta, E. Vidal, A. Sanchis, and J. Vilar, "Pattern recognition approaches for speech-to-speech translation," in *Cybernetic and Systems*, vol. 35, 2004, pp. 3–17.

[9] H. Mangu, E. Brill, and A. Stolcke, "Finding consensus in speech recognition: Word error minimization and other applications of confusion networks," in *CSL*, vol. 14, 2000, pp. 373–400.

[10] P. Clarkson and A. J. Robinson, "Language model adaptation using mixtures and an exponentially decaying cache," in *ICASSP*, vol. 2, 1997, pp. 799–802.

[11] S. Galliano, G. Gravier, and L. Chaubard, "The ESTER 2 Evaluation Campaign for the Rich Transcription of French Radio Broadcasts," in *ICSLP*, vol. 1, 2009, pp. 2583–2586.

[12] P. Deléglise, Y. Estève, and S. Meignier, "Improvements to the LIUM French ASR system based on CMU Sphinx: what helps to significantly reduce the word error rate?" in *ICSLP*, 2009, pp. 2123–2126.

[13] J. Civera, J. Vilar, E. Cubel, A. Lagarda, S. Barrachina, F. Casacuberta, E. Vidal, D. Picó, and J. González, "A syntactic pattern recognition approach to computer assisted translation," in *SSPR*, vol. 3138, 2004, pp. 207–215.

[14] E. Cubel, J. Civera, J. Vilar, A. Lagarda, S. Barrachina, E. Vidal, F. Casacuberta, D. Picó, J. González, and L. Rodrìguez, "A syntactic pattern recognition approach to computer assisted translation," in *ECAI04*, 2004, pp. 586–590.

[15] M. E. Wood and E. Lewis, "Windmill - the use of a parsing algorithm to produce predictions for disabled persons," in *Autumn Conference on Speech and Hearing*, vol. 18, 1996, pp. 315–322.

[16] P. Cardinal, G. Boulianne, M. Comeau, and M. Boisvert, "Real-Time Correction of Closed-Captions," in *ACL*, 2007, pp. 113–116.