Time- and Acoustic-Mediated Alignment Algorithms for Speech Recognition Evaluation

Simon Dobrišek, France Mihelič

Faculty of Electrical Engineering, Ljubljana University, Ljubljana, Slovenia
simon.dobrisek@fe.uni-lj.si

Abstract

The paper investigates the time- and acoustic-mediated alignment algorithms that can be used for better speech recognition evaluation. The edit-cost function, which weights the cost of speech unit matches, substitutions, deletions and insertions, is defined as a function of timed symbols or even as a function of speech signal segments. The algorithms are compared using several classical statistical measures of different types that are derived from speech recognition confusion matrices and are normally used to measure the agreement between different classifications of the same set of objects. These measures provide a reasonable indication that the investigated algorithms provide more relevant speech recognition error statistics than the algorithms that are commonly used for this purpose.

Index Terms: speech recognition evaluation, transcription alignment, time-mediated alignment, acoustic-mediated alignment, string edit distance

1. Introduction

The evaluation of speech recognition systems relies on the generally accepted scoring method that consists of comparing a reference transcription, which is assumed to accurately represent what the speaker said, with a hypothesis transcription representing the systems's hypothesis as to what the speaker said. The comparison is performed using variations of the classical alignment algorithm that computes the weighted edit distance between the two transcriptions. The algorithm produces the optimum alignment between speech units in the two transcriptions and determines the minimum-cost sequence of the basic edit operations needed to transform one transcription into the other. The basic edit operations are then tallied as recognition errors in terms of transcription unit substitutions, deletions and insertions [8, 6].

The optimum alignment is usually assigned so as to minimize the total recognition error rate (TER), which is the sum of the insertion, deletion and substitution errors, divided by the number of speech units in the reference transcriptions. The minimum number of substitutions, deletions and insertions needed to transform one transcription into the other is known as the Levenshtein distance. An important feature of this distance model is that multiple different optimum alignments may exist for the same pair of transcriptions with the same distance value [9]. This disturbing fact reduces the diagnostic value of examining the resulting error statistics, and it can even hinder the understanding of the system's failure mechanisms. A typical example that illustrates this problem is shown in Fig. 1.

There were several efforts to overcome this drawback of the Levenshtein-based alignment. The improvements are mainly based on weighting the costs of the basic edit operations that transforms one transcription into the other. Most of such improvements were proposed by the researchers from the speech group at NIST and are implemented as part of the NIST Scoring Toolkit (SCTK) [11].

```
REF: BASING IT      ON CERTAIN ITEMS   UH OVER THIS
HYP: BASING UM CERTAIN    ITEM     A HALF   OR THIS
ERR:         S       S       S     S    S    S

REF: BASING IT ON CERTAIN ITEMS  -   UH OVER THIS
HYP: BASING -- UM CERTAIN  ITEM  A HALF   OR THIS
ERR:         D  S             S  I    S    S
```

Figure 1: Two different Levenshtein-based alignments of the same pair of transcriptions with the same number of identified errors. Expert speech scientists would most probably prefer the bottom alignment that seems to be more in accordance with the phonological word similarities.

In any case, variations of the basic alignment algorithm differ mainly in the definition of the basic edit-cost function. The name of the alignment algorithm is usually even given according to the properties of the edit-cost function. For instance, if the edit-cost function is a function of words, phones, phonological features, time, or acoustic features, the alignment algorithm is called *word-mediated*, *phone-mediated*, *time-mediated*, or even *acoustic-mediated*. In this paper we investigate the advantages of the time-mediated alignment and propose an algorithm for the acoustic-mediated alignment for a better speech recognition evaluation.

Different variations of the alignment algorithm produce different resulting speech recognition error statistics, and determining which of them are more correct is not a trivial task [5]. Usually, it is not feasible to manually examine all the alignments and to make a decision about which of them are more or less correct. For this purpose we exploited our previous work, where we investigated whether any indication of which alignment algorithm produces more correct speech recognition error statistics can be drawn directly from the error statistics themselves [3]. On the basis of these statistical measures, we believe that the presented two algorithms provide more relevant speech recognition error statistics than the classical algorithms that are used for this purpose.

2. Time-mediated alignment algorithm

Speech transcriptions are usually given in the form of common strings, yet the time-marked speech transcriptions are also very common as speech recognition systems can normally produce not just hypothesis transcriptions, but also hypothesis speech segmentations. Time-marked reference transcriptions are often not available. However, they can always be obtained by using the so-called forced-alignment technique, which produces

reference speech segmentations from the given reference transcriptions.

Time-marked speech transcriptions are time-ordered finite sequences of timed symbols that are normally contiguous. We call such sequences contiguous timed strings. In our previous work, we proposed a general edit-distance model for contiguous and non-contiguous timed strings [4]. This model can be directly employed for the alignment of time-marked speech transcriptions.

Let us provide a brief overview of the timed edit-distance model with a focus on the contiguous timed strings. An algorithm for computing the the timed edit-distance function is derived from the recursion that is defined with the trace distance [9]. The main difference is that the basic edit operations are time dependent, and that we do not consider just a single timed null symbol. The basic edit operations that are called timed insertions and timed deletions are considered to be substitutions of timed symbols, assumed to be deleted or inserted, and timed null symbols that are positioned between two subsequent timed symbols in the opposite string.

Let $\mathbf{a} = (a, s, e)$ denote a timed symbol, where a is a symbol taken from a finite alphabet Σ and $s, e \in \mathbb{R}$ are the start and end times of the time interval when the symbol a is available. Let then Γ be the set of all the possible finite-duration timed symbols. Timed strings are time-ordered sequences of timed symbols, where the time-order is defined as the non-decreasing order of the middle times of the symbols in a sequence. Let $\mathbf{a}^n = \mathbf{a}_1 \cdots \mathbf{a}_i \cdots \mathbf{a}_n$ denote such a timed string, where n is the number of timed symbols. Note that the notation used does not distinguish between a timed string of length one and a single timed symbol, i.e., $\mathbf{a}^1 = \mathbf{a}_1$. The timed strings, for which it holds that the end time of $\mathbf{a}_i$ is equal to the start time of $\mathbf{a}_{i+1}$ for all $i = 1, \cdots, n-1$, are called *contiguous*. Let then Γ^+ denote the set of all the possible finite-length contiguous timed strings.

The concept of the edit distance between two timed strings requires the definition of an empty timed string. Let $\boldsymbol{\epsilon}$ denote an empty timed string that is defined as the identity element with the timed-string concatenation operation, i.e., $\boldsymbol{\epsilon}\ \mathbf{a}^n = \mathbf{a}^n \boldsymbol{\epsilon} = \mathbf{a}^n$ for all $\mathbf{a}^n \in \Gamma^+$.

At first glance, one could assume that there should be only one such empty timed string, and, that there is no need for it to be time dependent. On the other hand, an empty timed string can be seen rather as a time interval when no symbol is available. An empty timed string can thus be defined as a special timed string that comprises a timed null symbol. The proposed edit-distance model is based on this assumption and the results of the experiments presented later in the paper indicate that this may be an improvement over the use of the single time-independent empty string.

Let then ε be the null symbol that is associated with two time values. A timed null symbol is then denoted by $\boldsymbol{\epsilon} = (\varepsilon, s, e)$, where $s, e \in \mathbb{R}$ are the start and end times of the time interval when no symbol is available. Note that, in general, the timed null symbol can have non-zero duration. As the notation used does not distinguish between a single timed null symbol and a timed string that comprises only one such unit, the symbol $\boldsymbol{\epsilon}$ also denotes an empty timed string. Let Υ denote the set of all the possible empty timed strings.

The edit distance for the timed strings is defined by a pair $(\{\Gamma \cup \Upsilon\}, c)$, where $c : E \to \mathbb{R}_0^+$ is a timed edit-cost function that assigns non-negative real numbers to the timed edit operations in $E = E_s \cup E_d \cup E_i$, where $E_s = \Gamma \times \Gamma$ is a set of timed substitution operations, $E_d = \Gamma \times \Upsilon$ is a set of timed deletion operations, and $E_i = \Upsilon \times \Gamma$ is a set of timed insertion operations. Each such pair $(\{\Gamma \cup \Upsilon\}, c)$ induces a function $d : \{\Gamma^+ \cup \Upsilon\} \times \{\Gamma^+ \cup \Upsilon\} \to \mathbb{R}_0^+$ that maps a pair of timed strings into a non-negative real number. The function d is then defined as the minimum sum of the edit costs of the timed edit operations that, step by step, transform one timed string into another.

The computation of the function d is derived from the recursion that is defined with the simple trace distance. Let us define the recursion on the two timed strings $\mathbf{a}^n \in \Gamma^n$ and $\mathbf{b}^m \in \Gamma^m$. The recursion is defined as follows:

$$\begin{aligned} \mathrm{d}_{0,0} &= c(\boldsymbol{\epsilon}_0\,,\,\boldsymbol{\zeta}_0) \\ \mathrm{d}_{\mathrm{i},0} &= \mathrm{d}_{\mathrm{i}-1,0} + c(\mathbf{a}_i\,,\,\boldsymbol{\zeta}_0) \\ \mathrm{d}_{0,\mathrm{j}} &= \mathrm{d}_{0,\mathrm{j}-1} + c(\boldsymbol{\epsilon}_0\,,\,\mathbf{b}_j) \\ \mathrm{d}_{\mathrm{i},\mathrm{j}} &= \min \left\{ \begin{array}{lcl} \mathrm{d}_{\mathrm{i}-1,\mathrm{j}-1} & + & c(\mathbf{a}_i\,,\,\mathbf{b}_j)\,, \\ \mathrm{d}_{\mathrm{i}-1,\mathrm{j}} & + & c(\mathbf{a}_i\,,\,\boldsymbol{\zeta}_j)\,, \\ \mathrm{d}_{\mathrm{i},\mathrm{j}-1} & + & c(\boldsymbol{\epsilon}_i\,,\,\mathbf{b}_j) \end{array} \right\}, \end{aligned} \tag{1}$$

where $\mathrm{d}_{\mathrm{i},\mathrm{j}} = d(\mathbf{a}^i, \mathbf{b}^j)$ for $i = 0, \cdots, n$ and $j = 0, \cdots, m$ denotes the distance between the two timed substrings $\mathbf{a}^i$ and $\mathbf{b}^j$ of the two timed strings $\mathbf{a}^n$ and $\mathbf{b}^m$. The symbol $\boldsymbol{\zeta}_j$ denotes the timed null symbol that is positioned (squeezed) between $\mathbf{b}_j$ and $\mathbf{b}_{j+1}$ in the same way as the timed null symbol $\boldsymbol{\epsilon}_i$ is positioned (squeezed) between $\mathbf{a}_i$ and $\mathbf{a}_{i+1}$.

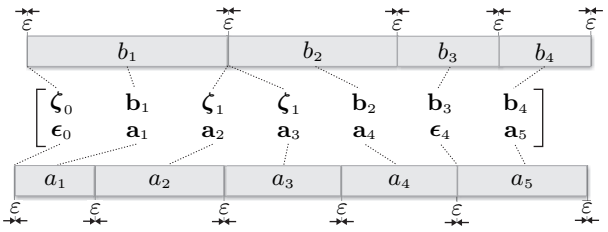

Figure 2: A graphical presentation of the optimum alignment between the two contiguous timed strings $\mathbf{a}^5$ into $\mathbf{b}^4$.

The recursion starts with a pair of possibly different empty timed strings $\mathbf{a}^0 = \boldsymbol{\epsilon}_0$ and $\mathbf{b}^0 = \boldsymbol{\zeta}_0$. The initial cost $c(\boldsymbol{\epsilon}_0\,,\,\boldsymbol{\zeta}_0)$ is thus not necessarily zero. This cost can be seen as a cost for the initial time synchronization of the two time strings being transformed to each other. The recursion is then calculated for all $i = 1, \cdots, n$ and $j = 1, \cdots, m$ and the final $\mathrm{d}_{\mathrm{n},\mathrm{m}} = d(\mathbf{a}^n, \mathbf{b}^m)$ is the timed string-edit distance between $\mathbf{a}^n, \mathbf{b}^m$. The minimum-cost sequence of edit operations that transforms $\mathbf{a}^n$ into $\mathbf{b}^m$ can be identified at the end of the recursion by starting at $\mathrm{d}_{\mathrm{n},\mathrm{m}}$ and tracing the local minima back to $\mathrm{d}_{0,0}$. Fig. 1 shows the optimum timed alignment of the two timed strings $\mathbf{a}^5$ into $\mathbf{b}^4$.

The crucial parts of the recursion in Eqs. (1) are the costs of the edit operations, defined by the edit-cost function c. We investigated the timed edit-cost function that is defined as a weighted sum of a symbol-dissimilarity and time distance between the two timed symbols [4]. For the symbol dissimilarity we can use any of the usual cost edit-cost functions that are used for such alignments. The time distance is a function of a pair of time stamps and for this function we can use any of the distance metrics that are defined on $\mathbb{R} \times \mathbb{R}$, among them are the Manhattan, Euclidian, Chebyshev and others

3. Acoustic-mediated alignment algorithm

The acoustic-mediated alignment algorithm that we propose in this paper is derived from the time-mediated alignment algorithm above. The main difference is in the edit-cost function that is part of the recursion in Eqs. (1). The costs of the timed edit operations should be related to some dissimilarity measure between the two timed symbols being compared. Our timed symbols hold information about speech segmentation and, if the speech signal that is transcribed by the timed string is available, one can define this dissimilarity as a function of speech segments. For this purpose we can use the DTW distance measure [10] on sequences of the acoustic features vectors derived from the corresponding speech signal segments. This measure can be used for weighting substitutions; however, it cannot be used for insertions and deletions as the DTW distance measure does not support null units, i.e., null speech segments.

By considering the properties of the edit-distance model in general, we define the acoustic edit-cost function as follows. Let $\mathbf{X}_i = \{\mathbf{x}_{i1}, \cdots, \mathbf{x}_{in}\}$ denote a sequence of acoustic feature vectors corresponding to the timed symbol $\mathbf{a}_i$, $\mathbf{Y}_j = \{\mathbf{y}_{j1}, \cdots, \mathbf{y}_{jm}\}$ a sequence of acoustic feature vectors corresponding to the timed symbol $\mathbf{b}_j$, and $\{\,\}$ an empty sequence of feature vectors corresponding to all the possible timed null symbols $\boldsymbol{\epsilon}_i$ and $\boldsymbol{\zeta}_j$. Let then c_t denote the acoustic distance between two sequences of features vectors. The acoustic distance between two non-empty sequences $c_t(\mathbf{X}_i, \mathbf{Y}_j)$ is defined as the normalized DTW distance

$$c_t(\mathbf{X}_i, \mathbf{Y}_j) = \frac{\mathrm{D}(\mathbf{X}_i, \mathbf{Y}_j)}{||\mathbf{X}_i||_\Delta \cdot ||\mathbf{Y}_j||_\Delta \cdot \max\{n, m\}} \,,$$

where D denotes the usual DTW distance measure with the basic symmetric local constraints, and $||\cdot||_\Delta$ denote the maximum norm of the feature vectors in the given sequence. The normalizing denominator ensures that c_t is bounded on the interval $[0, 1]$. Let then the acoustic distance c_t between a non-empty and an empty sequence of feature vectors (and vice versa) equal 1, and between two empty sequences equal 0. The edit cost function c in Eqs. (1) is then defined as

$$\begin{array}{rcl} c(\boldsymbol{\epsilon}_0, \boldsymbol{\zeta}_0) & = & 0\,, \\ c(\mathbf{a}_i, \mathbf{b}_j) & = & \left\{ \begin{array}{ll} c_t(\mathbf{X}_i, \mathbf{Y}_j)\, c_{id} & ;\ s_i = s_j\,, \\ c_t(\mathbf{X}_i, \mathbf{Y}_j)\, (c_s - c_{id}) + c_{id} & ;\ s_i \neq s_j \end{array} \right. , \\ c(\mathbf{a}_i, \boldsymbol{\zeta}_j) & = & c_d\,, \\ c(\boldsymbol{\epsilon}_i, \mathbf{b}_j) & = & c_i\,, \end{array}$$

where c_s, c_d, and c_i denote the cost for substituting, deleting or inserting the symbol part of the corresponding timed symbols, c_{id} denote $\min\{c_d, c_i\}$, and s_i and s_j denote the symbol part of the timed symbols $\mathbf{a}_i$ and $\mathbf{b}_j$, respectively.

4. Comparison of the alignment algorithms

The presented alignment algorithms were evaluated by comparing the speech recognition error statistics that they provided when we used them for speech recognition evaluations. Speech recognition error statistics are normally given in the form of confusion matrices and some indication as to which alignment algorithm produces more relevant error statistics may be drawn directly from these matrices. We investigated many different evaluation functions that are well-known in statistics and are commonly used for measuring the agreements between different classifications of the same set of objects [3].

Fig. 3 illustrates our evaluation problem. Two different alignment algorithms calculated similar total phone-recognition

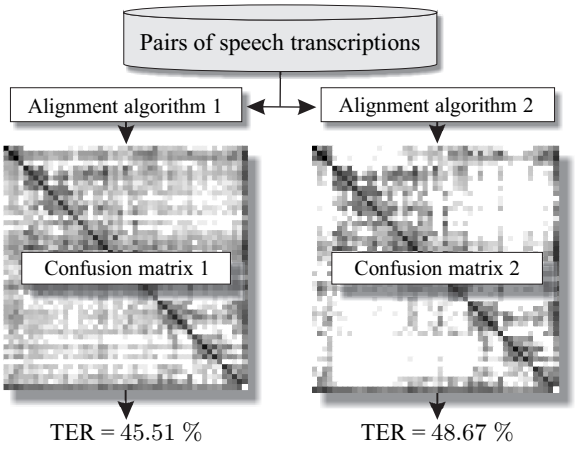

Figure 3: The two graphically represented confusion matrices were obtained using two different alignment algorithms from the same pairs of reference and hypothesis phonetic transcriptions. An indication as to which alignment algorithm produces more correct error statistics can be drawn directly from the confusion matrices.

error rates from the same pairs of phonetic transcriptions, however, they produced two rather different confusion matrices. From the illustration it is clear that in the left-hand matrix the phone-recognitions errors are distributed more randomly over the pairs of phonetic units than is the case with the right-hand matrix. Considering the nature of human speech production and perception, one would intuitively expect that the less random distribution of errors should be the more correct[2].

On the other hand, the first scoring algorithm detected a considerably lower total number of errors (TER $= 45.51\%$) than the second one (TER $= 48.67\%$). It is clear that only from the values of TER we cannot make any obvious assumption about which of the two algorithms produces the more correct error classification. More precisely, a smaller number of detected errors does not necessarily indicate that the corresponding algorithm detects errors more correctly.

Many different statistical measures were proposed to measure the degree of association between two categorial variables and most of them are derived from the contingency table [7]. These measures include the chi-squared, phi-squared and G-test (GT), Cramer's V (CV) and the lambda (λ) statistics, normalized mutual information (NMI), etc. The speech recognition confusion matrices are special examples of contingency tables, where the two corresponding variables range over the same set of categories (i.e., speech units). One variable is associated with the reference transcriptions and the other with the hypothesis ones. The null symbol, which is used for the representation of deletion and insertion errors, is also considered as a category by itself in the same way as all the other speech units.

Another group of measures is normally used for measuring the agreement between two observers that make statistical decisions on a given hypothesis. One of them is considered as a reference observer and the other can make false-positive and false-negative decisions. The results of their decisions are recorded and analysed using the usual 2×2 contingency tables. The hypothesis that we considered in our evaluation experiment is normally used for the strict comparison of two speech classifications, i.e., the measure of agreement between two classification takes its maximum value if and only if the total number

		alignment algorithm				
		LBA	NWLA	NTMA	TMA	AMA
I	κ	0.653	0.655	0.612	0.656	0.655
	CV	0.657	0.666	0.631	0.666	0.665
	λ	0.641	0.645	0.601	0.647	0.646
	NMI	0.610	0.623	0.607	**0.645**	**0.647**
	GT	143455	147407	144500	**152809**	**152845**
II	FM	0.664	0.666	0.625	0.667	0.666
	J	0.497	0.500	0.454	0.500	0.500
	AR	0.657	0.659	0.617	0.660	0.659
	YY	0.887	0.888	0.871	0.888	0.888
	TER	34.57	34.58	39.22	34.58	34.62

Table 1: The values of all the considered statistical measures (given in abbreviations) obtained by different aligning algorithms on the TIMIT phone recognition evaluation.

		alignment algorithm			
		LBA	NWLA	NTMA	TMA
I	κ	0.627	0.629	0.552	0,629
	CV	0.499	0.503	0.504	**0,516**
	λ	0.628	0.629	0.538	0,630
	NMI	0.658	0.667	0.625	**0,683**
	GT	3764758	3827496	3644208	**3926353**
II	FM	0.632	0.634	0.560	0,634
	J	0.462	0.465	0.389	0,465
	AR	0.632	0.634	0.560	0,634
	YY	0.993	0.993	0.991	0,993
	TER	38.90	38.92	48.48	39.02

Table 2: The values of all the considered statistical evaluation measures obtained by different aligning algorithms on the AMI Meeting Corpus word-recognition evaluation.

of errors is zero. Many different such measures are defined as functions of the number of all the four possible decisions [7]. We focused on the the Fowlkes-Mallows's (FA), Jaccard (J), Adjusted Rand (AR), and Yules Y (YY) indexes. All these indexes take their maximum value of 1 when the number of false-positive and false-negative decisions both equal 0, i.e., the two classifications are in complete agreement. On the other hand, if the number of true-positive and true-negative decisions both equal 0, then the values of these indexes are equal to or less than zero, i.e., the two classifications are in complete disagreement [3].

5. Experimental results

Two different groups of experiments for two different speech recognition evaluation problems were conducted. For the phone recognition evaluation experiment, the timed-marked reference and hypothesis phonetic transcriptions of the test part of the TIMIT database was used. The hypothesis phonetic transcriptions were obtained using our own simple HTK-based phone recognizer. For the word recognition evaluation experiments, the time-marked reference and hypothesis word transcriptions of the AMI Meeting Corpus[1] were used. The hypothesis transcriptions of this corpus were obtained from the AMI ASR group. The results are given in Tab. 1 and Tab.2.

In both tables, the basic Levenshtein-based alignment algorithm, which gives the minimum possible total error rate, is denoted as LBA. The basic NIST SCTK scoring algorithm, which assigns a cost value of 4 to substitutions and 3 to insertions and deletions, is denoted as NLWA. The NIST SCTK time-mediated alignment algorithm is denoted as NTMA. This algorithm differs from ours in the way the basic timed edit-cost function is defined [11]. The proposed time- and acoustic-mediated alignment algorithms are denoted as TMA and AMA. In both algorithms, we used the same symbol edit costs that are defined with the NLWA algorithm. The weight in the TMA edit-cost function was set to 0.5 for the phone recognition experiment and to 0.9 for the word recognition experiment. The AMA algorithm was not evaluated with the AMI Meeting Corpus as its original speech signals were not available to us.

According to our assumption about the used statistical measures [3], the results given in bold give a good indication that in both experiments the presented two algorithms provided more relevant speech recognition error statistics than the NIST algorithms. Namely, the differences in the values of NMI and GT are significant and provide a good indication that our two algorithms generated less randomly-distributed recognition error statistics. The differences between the values of the other measures seem to be not very significant, however, the values of our two algorithms are never worse than the values of the NIST algorithms.

6. Conclusions

In this paper several variations of the alignment algorithms are discussed and compared. An indication is given that the time- and acoustic-mediated algorithms provide more relevant error statistics which improves the diagnostic value of the obtained speech recognition evaluation results.

7. References

[1] Ashby, S., Bourban, S., Carletta, J., Flynn, M., Guillemot, M., Hain, T., Kadlec, J., Karaiskos, V., Kraaij, W., Kronenthal, M., Lathoud, G., Lincoln, M., Lisowska, A., McCowan, I., Post, W., Reidsma, D., Wellner, P., "The AMI Meeting Corpus", In: Measuring Behavior 2005 Proceedings Book, Wageningen, NL, 2005.

[2] Cutler, A., Weber, A., Smiths, R., and Cooper, N., "Patterns of English phoneme confusions by native and non-native listeners", Journal of the Acoustical Society of America, 116(6):3668–3678, 2004.

[3] Dobrišek, S., Mihelič, F., "Criteria for the Evaluation of Automated Speech-Recognition Scoring Algorithms", Electrotechnical Review, 75(4):229–234, 2008.

[4] Dobrišek, S., Žibert, J., Pavešić, N., Mihelič, F., "An Edit-distance Model for the Approximate Matching of Timed Strings", IEEE TPAMI, 31(4):736–741, 2009.

[5] Doddington, D., "Word alignment issues in ASR scoring", Proceedings of ASRU'03, 630–633, 2003.

[6] Gibbon, D., Moore, R., and Winski, R., [Eds], "Handbook of Standards and Resources for Spoken Language Systems", Mouton de Gruyter, Walter de Gruyter Publishers, Berlin, 1997.

[7] Hill, T., and Lewicki, P., STATISTICS - Methods and Applications, StatSoft, Tulsa, OK, 2007.

[8] Fisher, W. M. and Fiscus, J. G., "Better Alignment Procedures For Speech Recognition Evaluation", Proceedings of ICASSP'93, 2:59–62, 1993.

[9] Kruskal, J. B., and Sankoff, D., "An Antology of Algorithms and Concepts for Sequence Comparisons", Time Warps, String Edits and Macromolecules: The Theory and practice of Sequence Comparison, CSLI Publications, 256–310, 1999.

[10] C. S. Myers, C., S., Rabiner, L., R., "A comparative study of several dynamic time-warping algorithms for connected word recognition", The Bell System Technical Journal, 60(7):1389–1409, 1981.

[11] "The NIST Speech Recognition Scoring Toolkit (SCTK) Version 2.4.0", [Web page], `http://www.nist.gov/speech/tools`, The NIST Speech Group. National Institute of Standards and Technology (NIST), USA. [Accessed March, 2011].