NISTIR 5129

Methods for Evaluating the Performance of Systems Intended to Recognize Characters from Image Data Scanned from Forms

Michael D. Garris

U.S. DEPARTMENT OF COMMERCE
Technology Administration
National Institute of Standards
and Technology
Computer Systems Laboratory
Advanced Systems Division
Gaithersburg, MD 20899

February 1993

U.S. DEPARTMENT OF COMMERCE
Ronald H. Brown, Secretary

**NATIONAL INSTITUTE OF STANDARDS
AND TECHNOLOGY**
Raymond G. Kammer, Acting Director

Methods for Evaluating the Performance of Systems Intended to Recognize Characters from Image Data Scanned from Forms

Michael D. Garris
National Institute of Standards and Technology
Gaithersburg, MD 20899

1. Introduction

The concepts presented in this paper were developed to establish a uniform method of evaluating the recognition of optical character readers used to process the information on forms which receive information as a bit stream directly or indirectly from scanners. Many large data entry systems are being designed to collect data from specified areas of forms, some of which may be multipart and completed with machine-printed or hand-printed characters. As this technology continues to advance, the number of commercially available products is increasing. Multiple products are emerging, all of which are designed to solve optical character recognition (OCR) applications. Improved recognition algorithms have enabled the accuracy of these products to steadily increase, but each product is based on a different, often proprietary, set of algorithms. This presents potential users of OCR technology with many different choices and options and leads to a series of significant questions: How does a person determine when the technology has matured enough to make it economically advantageous to deploy? How does a potential user determine which product is best for his or her specific needs? How can a system developer, who has the ability to choose from a large variety of diverse algorithmic approaches, intelligently choose and then track progress when developing OCR systems? The answer to these questions lies in objective system performance measurement. This is the motivation behind this paper.

Section 2 discusses referenced image databases and how they are used to test recognition systems. Section 3 introduces the general principles of form-based scoring. Section 4 demonstrates how form-based scoring is applied to a specific OCR application, and fundamental ideas related to dynamic string alignment are presented in Appendix A.

2. Testing Recognition Systems Using Referenced Image Databases

Application requirements germane to a specific OCR problem are embodied in a representative set of referenced images. Associated with each reference image is the ASCII textual information that is to be recognized in the image. The reference information in these databases serves as ground truth for measuring recognition performance. The images are presented to a recognition system, and the system's results are returned. This includes hypothesized text of what the system located and recognized. NIST has implemented these methods in the form of a Scoring Package that reconciles the hypothesized text with the reference text, accumulating statistics used to compute performance measures.[1] The NIST Scoring Package was used to compute the results from the First Census Optical Character Recognition System Conference sponsored by the Bureau of the Census and hosted at NIST.[2] Figure 1 illustrates the use of referenced images and the NIST Scoring Package to assess the performance of a recognition system.

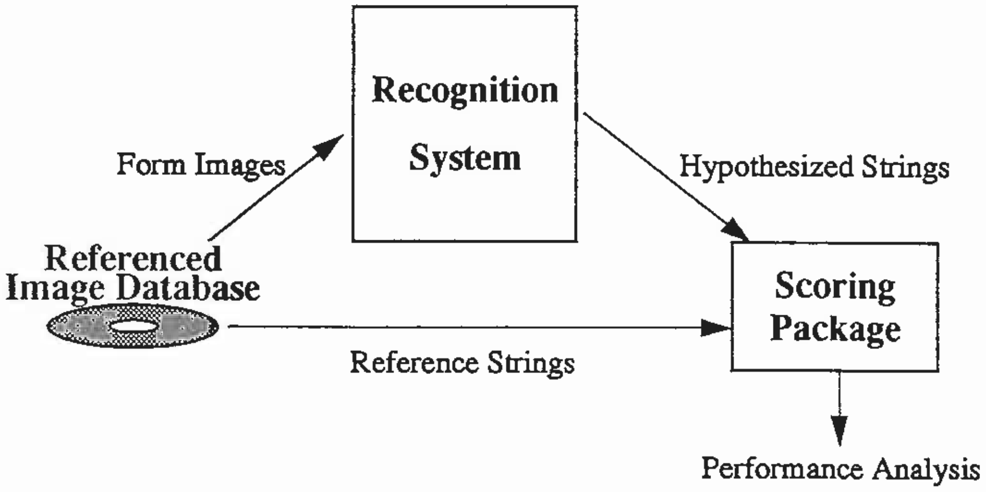


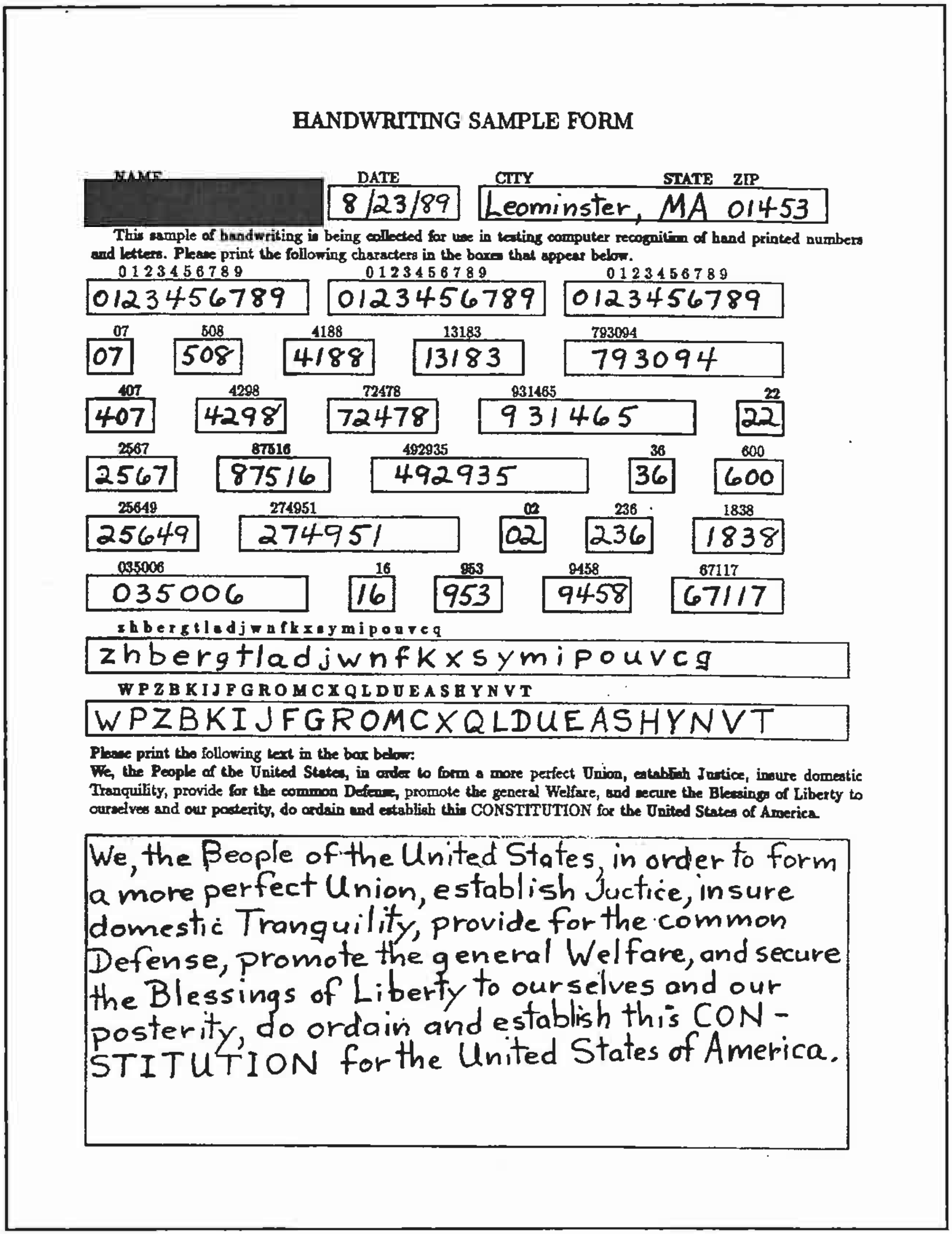
Figure 1: Testing paradigm for recognition systems using referenced images.

The model in Figure 1 has several advantages. First, knowledge of the internal details of a system being tested is not required. This is critical when testing systems comprised of proprietary functional components. Second, the performance measures are computed in an automated way without any human inspection. This is extremely important when assessing the performance of OCR technology, especially large-scale character recognition systems. An example is the NIST massively parallel model recognition system whose character recognition component is capable of classifying character images up to 1000 characters per second[3]. This system is capable of processing 2,100 pages of forms containing 130 hand-printed digits per form for a total of 273,000 digits in approximately 4 hours. The visual inspection of the system output from a single 4-hour processing session took a technician 6 months. In order to conduct tests in a reasonable amount of time, the compiling and computing of performance measures must be automated.

Separate tests must be performed for each application of OCR technology because the results of tests performed on one application cannot be assumed to apply to another application. Using the system testing paradigm in Figure 1, potential users of character recognition technology must design a collection of referenced images representative of their specific needs. The set of images can then be presented to different candidate systems, and using these methods, performance measures can be computed from the output of each system for the purpose of system comparison. Likewise, a system developer can take a set of referenced images and present them to several variations of a single system. For example, one system configuration may use algorithm A for character segmentation, whereas another system configuration may use algorithm B. By presenting the same set of referenced images to both system configurations, performance measures can be computed and used to compare the two algorithms within the context of a fully operational system.

NIST has produced three referenced image databases of digitized forms which are available to the public and distributed through NIST's Standard Reference Data Division on CD-ROM. *NIST Special Database 1* (SD1)[4] contains 2,100 digitized pages of a hand-print collection form completed by 2,100 different writers geographically distributed across the United States. Figure 2 shows an example of one of these forms. Each full-page image in the database is a form comprised of 33 entry fields. Each entry field is demarcated by a separate box on the form. These fields include 28 numeric fields totalling 130 hand-printed digits, 1 alphabetic field containing the 26 lower-case letters, 1 alphabetic field containing the 26 upper-case letters, and a text paragraph field containing the first sentence from the Preamble to the Constitution of the United States. *NIST Special Database 2* (SD2)[5] contains 5,590 digitized tax forms from the IRS 1040 Package X for the year 1988 completed with machine-print. These include Forms 1040, 2106, 2441, 4562, and 6251 together with Schedules A, B, C, D, E, F, and SE. *NIST Special Database 6* (SD6)[6] contains 5,595 digitized tax forms from the same list completed with hand-print. The information provided on these tax forms has been generated by a computer and does not represent real people or real tax data.

Two other referenced databases are available to the public from NIST. They contain images of isolated characters that are useful for testing in isolation the character classification components of full-scale recognition systems. *NIST Special Database 3* (SD3)[7] contains 313,389 images of segmented characters from the 2,100 writers in SD1. SD3 contains 223,125 digits, 44,951 upper-case letters, and 45,313 lower-case letters. These images have been verified to contain correctly segmented characters and do not include images of split or merged characters. Associated with every character image in this database is a reference value specifying the class of the character in the image. A second character image database, *NIST Special Database 7* (SD7)[8], was intended to be used primarily for testing hand-print character classifiers. SD7 contains hand-print from 500 writers and has approximately 83,000 isolated character images including 59,000 digits and 24,000 upper-case and lower-case letters. Because SD7 was a testing database, the reference classifications for the character images are distributed on floppy disk separately from the character images which are distributed on CD-ROM.



HANDWRITING SAMPLE FORM

NAME [redacted] DATE 8/23/89 CITY Leominster, STATE MA ZIP 01453

This sample of handwriting is being collected for use in testing computer recognition of hand printed numbers and letters. Please print the following characters in the boxes that appear below.

0123456789	0123456789	0123456789		
0123456789	0123456789	0123456789		
07	508	4188	13183	793094
07	508	4188	13183	793094
407	4298	72478	931465	22
407	4298	72478	931465	22
2567	87516	492935	36	600
2567	87516	492935	36	600
25649	274951	02	236	1838
25649	274951	02	236	1838
035006	16	953	9458	67117
035006	16	953	9458	67117

zhbergtladjwnfkxsymipouvcq

zhbergtladjwnfKxsymipouvcg

WPZBKIJFGROMCXQLDUEASHYNVT

WPZBKIJFGROMCXQLDUEASHYNVT

Please print the following text in the box below:
We, the People of the United States, in order to form a more perfect Union, establish Justice, insure domestic Tranquility, provide for the common Defense, promote the general Welfare, and secure the Blessings of Liberty to ourselves and our posterity, do ordain and establish this CONSTITUTION for the United States of America.

We, the People of the United States, in order to form a more perfect Union, establish Juctice, insure domestic Tranguility, provide for the common Defense, promote the general Welfare, and secure the Blessings of Liberty to ourselves and our posterity, do ordain and establish this CONSTITUTION for the United States of America.

Figure 2: Example of a form comprised of entry fields completed with hand-print.

3. Concepts of Form-Based Scoring

Form-based scoring methods have been developed to measure the performance of character recognition systems, and more specifically, automated form processing systems. Many large data entry systems must collect data from specified areas of forms, some of which may be multipart and completed with machine-printed or hand-printed characters. Figure 2 shows an image of a form from SD1 containing hand-print. The first entry field on the form has been blanked out to make the writer of the form anonymous. Other examples of forms include documents such as Census forms, IRS tax forms, credit card slips, checks, etc.

This section describes the procedures necessary to assess the performance of automated form processing systems. Choosing an OCR application is the first step required to design a recognition system test. Once an application is selected, the recognition tasks embodied in the application and the interactions between tasks that impact system performance are identified. Based on these tasks and their interaction, a scoring flow is derived. Scoring accumulators designed to capture system performance statistics are defined within the scoring flow. Finally, recognition performance measures that use the scoring accumulators as input are defined.

3.1 Performance Assessment Based on Target Applications

A recognition system test is developed by first targeting a specific OCR application. Target applications are defined by two principal components. The first component is a testing set comprised of imaged forms. These images must be representative of the types of forms and the types of information to be captured by the recognition system. The second component is a collection of reference system responses. Typically, the types of forms included in the testing set dictate which form processing tasks are embodied within the target application. A separate reference response must be recorded for each of the tasks to be performed on each form in the testing set. In general, a target application that includes form identification requires a reference form identification for each form in the testing set, a target application that includes field identification requires a reference field identification for each field in the testing set, a target application that includes field recognition requires a reference field value for each field in the testing set, and a target application that includes character recognition requires a reference character value for each character in the testing set. These tasks are described below.

Together, the form images and the reference responses represent a referenced image database like those discussed in Section 2. Hypothesized system responses are generated by presenting the imaged forms to the recognition system. System performance is then measured by compiling statistics based on comparing the hypothesized system responses to the reference system responses. By defining a target application in this manner, performance assessment can be viewed as a test. The imaged forms represent the test's questions, the hypothesized system responses represent the student's answers, and the reference system responses represent the teacher's answer sheet.

3.2 Form Processing Tasks

Figure 3 illustrates four different form processing tasks addressed by this paper. These tasks include form identification, field identification, field recognition, and character recognition. Different target applications may require the scoring of other tasks, but these four tasks embody the primary functions that distinguish form processing from other OCR application domains such as reading free-formatted correspondence. Also notice that these tasks in no way limit the implementation of a form processing system by dictating a presumed set of algorithmic procedures. For example, traditional character recognition systems conduct character segmentation prior to character classification.[3][9] Methods of combining segmentation and classification into a single concurrent process have recently been developed.[10][11][12] Regardless of the algorithm used, both types of systems produce character classifications that can be analyzed and compared, and both systems can be analyzed based on their character recognition performance.

In general, the first step to processing a form requires proper identification of the form type. By establishing form identification as the first task, the methods of performance assessment presented in this paper do not address system issues such as pages missing from a multiple-page document, and other page handling issues. Based on the identified form type, fields may be located through the use of a spatial template. If fields cannot be unambiguously identified by position alone, then other contexts may be required such as reading the label printed on the form next to each field. This is referred to as field identification. Once a field has been located and identified, it then can be recognized. The task of reporting a single response for an entire field is referred to as field recognition, whereas the task of reporting single responses, one for each character in a field, is referred to as character recognition.

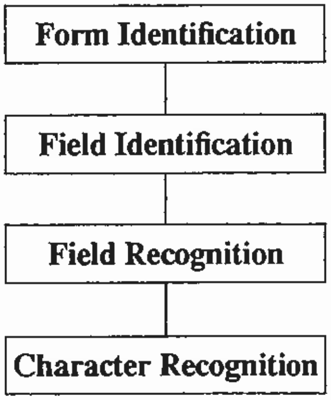


Figure 3: Four tasks of a generic form processing system.

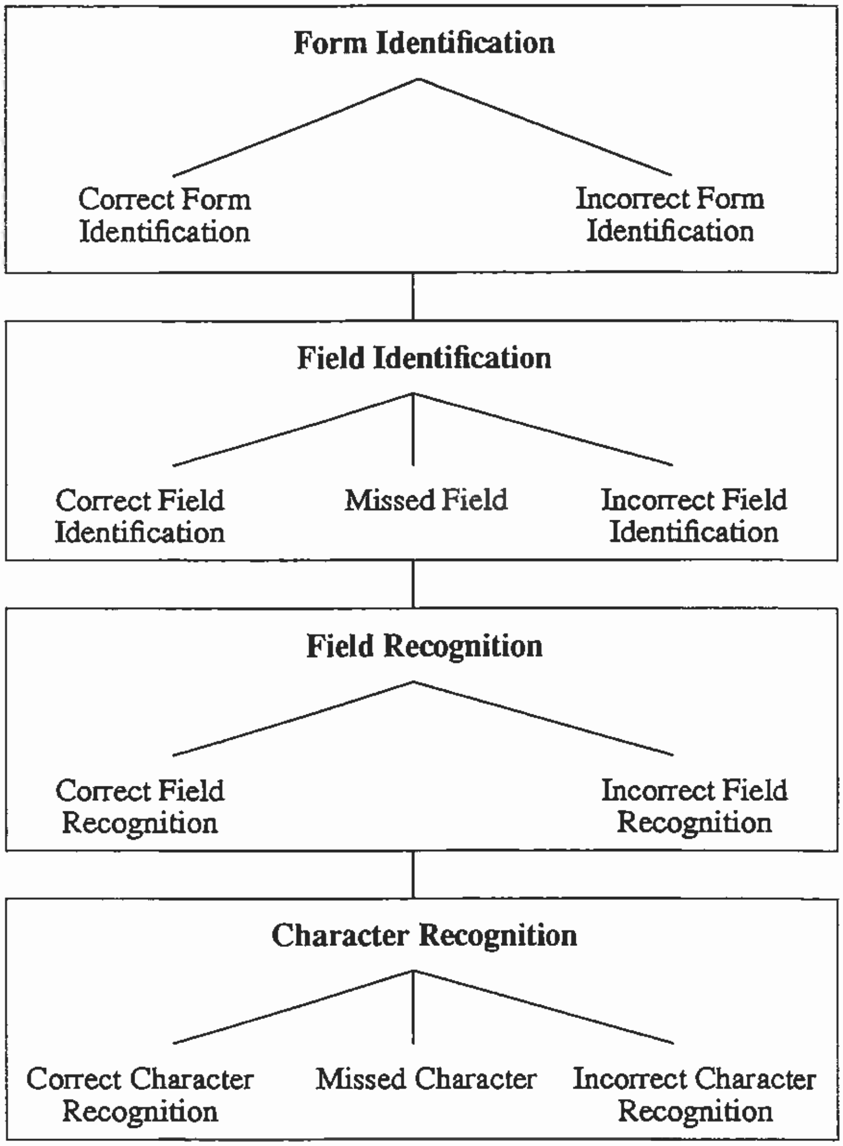

Form Identification
Correct Form Identification
Incorrect Form Identification
Field Identification
Correct Field Identification
Missed Field
Incorrect Field Identification
Field Recognition
Correct Field Recognition
Incorrect Field Recognition
Character Recognition
Correct Character Recognition
Missed Character
Incorrect Character Recognition

Figure 4: The possible outcomes resulting from each of the four form processing tasks.

A more detailed diagram of the form processing tasks is shown in Figure 4. This figure illustrates the possible outcomes resulting from each of the four tasks. Form identification can either result in a correctly identified form or an incorrectly identified form. Likewise, field identification can either result in a correctly identified field, an incorrectly identified field, or a missed field. Field recognition can result in an entire field being recognized correctly or incorrectly, while character recognition can result in a character being correctly recognized, incorrectly recognized, or missed. Incorrectly recognized characters include both substituted and inserted characters. Characters are frequently missed due to errors during segmentation. Performance measurements can be computed by compiling statistics at each of these possible outcomes.

Up to this point, the effects of system rejections on scoring have not been addressed. Systems have the potential to reject the outcomes from all four of the form processing tasks. This is illustrated in Figure 5. For example, a system may choose to reject the hypothesized form type assigned to a specific form image, or a system may choose to reject the hypothesized classification assigned to a segmented character image. Rejecting outcomes gives a system the ability to flag low confidence decisions as unknown, so that they may be verified by human inspection. Figure 6 illustrates how the performance of a recognition system's rejection decisions can be analyzed.

The first diagram measures the system's ability to perform the task, while the second diagram measures the system's ability to reject or accept system responses accurately. In order to conduct the analysis in Figure 5, the reference responses must correspond to the appropriate task outcome. The analysis represented in Figure 6 requires reference responses that correspond to appropriate system decisions to accept or reject identification or recognition results.

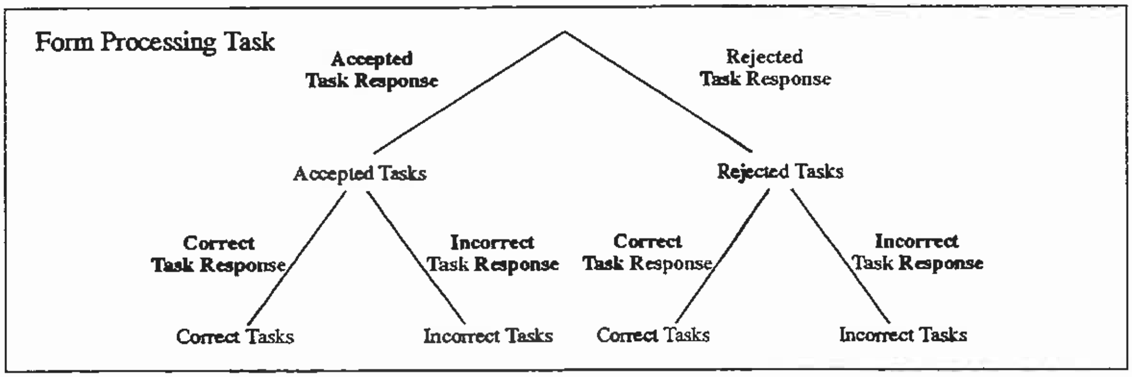


Figure 5: Measuring system task performance based on a system's decision to reject responses.

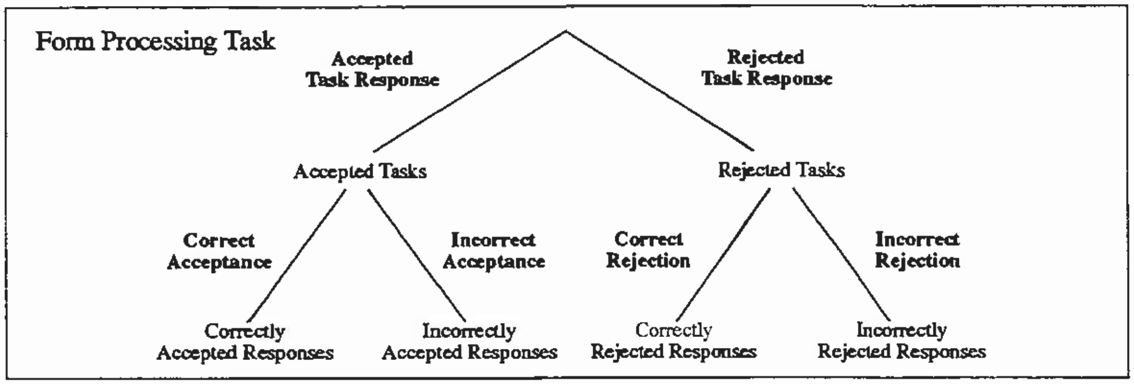


Figure 6: Measuring the performance of a system's rejection decisions.

3.3 Task Interactions and Scoring Flow

The diagrams in Figure 4, Figure 5, and Figure 6 should not be mistaken as a model for implementing form processing systems. They should be viewed as a flexible framework by which form processing systems can be analyzed and compared. A specific form processing application is assumed to contain one or more of the form processing tasks listed in Figure 3. Those tasks not included in the target application, for example a system may not conduct field identification, are not included in the analysis of recognition system performance. A scoring flow is thereby defined according to the presence of these tasks and their interactions between each other.

Task interactions of interest are those that impact system performance. For example, it may be determined that within an application a system rejection of a form's identification should result in all characters on the form being tallied as missed. In this case, a decision made within the form identification task influences performance within the character recognition task (characters are missed). A common task interaction is to analyze system responses at subsequent tasks only when the recognition system's response at the current task has been accepted by the system and the response is correct (equal to the reference system response). For example, only fields on forms that have been accepted and correctly identified should be analyzed at the field identification or recognition tasks. Scoring flows are defined through this process of identifying tasks and defining their performance-based interactions.

3.4 Scoring Accumulators and Performance Measures

Once the scoring flow has been defined, scoring variables must be defined in order to accumulate system performance statistics. Variable names are defined by one or more letters representing a type of recognition system decision and/or scoring outcome. These letters include recognition system decisions to accept (A) or reject (R) system results. They also include scoring outcomes that determine if the recognition system's decisions were correct (C) or incorrect (I) or whether information was missed (M) by the recognition system. Performance statistics may be accumulated at the form, field, and character levels and may be represented as the variable subscripts *form*, *field*, and *char* respectively. The form processing task contributing to a particular statistical accumulator is denoted by the variable's superscript. Statistics accumulated for forms identification may be denoted as *frmid*, field identification as *fldid*, field recognition as *fldrec*, and character recognition as *chrrec*. Other names for subscripts and superscript may be required depending on the target OCR application chosen.

Using this nomenclature, variable accumulators used to compute system performance measures can be defined. For example, the variable RC_{form}^{frmid} can be used to represent the total number of correctly identified forms rejected by the recognition system. Likewise, a variable representing the total number of characters missed during character recognition may be M_{char}^{chrrec}. Several other accumulators are also required for scoring. They include the total number of forms processed $total_{form}$, the total number of fields processed $total_{field}$, and the total number of reference characters on the forms processed $total_{refchr}$.

Different elements of recognition system performance are critical to different form processing applications. For example, one form processing application may require a very high acceptance rate providing a high level of automated throughput, and the application may be tolerant of a moderate level of error. Another form processing application may require very low error rates at the expense of a very low acceptance rate. The methods presented in this paper have been designed to accumulate performance statistics at a fundamental level. These statistics can then be combined and used in complex measures designed to capture the attributes of system performance relevant to a specific application. The performance measures discussed in this paper assess the accuracy of character recognition systems and do not have associated with them any direct notion of cost. Of course, scoring accumulators can be used as inputs to cost models, but the assessment of performance-based cost is not covered in this paper.

4. Case Study

This section demonstrates how form-based scoring can be applied to a specific application. By following the general guidelines presented above, an OCR application is described, the relevant recognition tasks are identified, interactions between the tasks are defined, scoring flows are derived, scoring accumulators are defined, and performance measures are presented. Based on the designed test, a recognition system is presented all the imaged forms in the testing set and all the system's responses for each of the tasks are recorded. The hypothesized responses are then entered into the scoring flow and compared against the reference system responses. Scoring accumulators are incremented according to all task interactions for all hypothesized responses. Once all hypothesized system responses have been scored, the accumulators are combined to compute system performance measures that are analyzed to assess the realized system performance.

4.1 The Application

The application studied in this example is the automatic processing of different structured form types, where each form contains multiple hand-printed character fields, fields containing box check marks, and fields containing signatures. The forms are structured so that the number, type, and location of each entry field remains consistent across all forms of the same type. The recognition system is required to recognize the type of each form processed and recognize every field on the form. If the field contains characters, it is referred to as a character-field, and the system is required to report recognition responses character by character. If the field contains non-character information, it is referred to as an icon-field, and the recognition system is required to determine if the field contains information or not. If the icon-field is a signature field, the recognition system is required to determine if the field has a signature present. If the icon-field is a box to be checked off, the recognition system is required to determine if the field has a check mark in it. Finally, the recognition system has the ability to reject form identifications, entire field responses, and individual character classifications.

4.2 Relevant Form Processing Tasks

This structured form processing application embodies three of the tasks listed in Figure 3. From the application description above, two of the tasks are obvious. They are form identification and character recognition. The task of field identification is not included because the location of entry fields is directly dependent on the identification of the form's type. This is true when, based on the type of the form, a spatial template can be applied to the image to isolate the entry fields in the image. The third task included indirectly in the application description is field recognition. The processing of icon-fields falls within this task, and character-field recognition responses can be inferred from the responses generated within the character recognition task. Please note that inferring character-field recognition responses from individual character recognition responses would be avoided if the application required character-field level responses to be reported separately. In this case, the system's character-field recognition responses would be matched directly to reference character-field recognition responses.

4.3 Task Interactions and Scoring Flows

With the three form processing tasks identified, the interactions between tasks must be defined and a scoring flow derived. Beginning with the form identification task, if the type of a form is correctly identified, then the form is tallied as correctly identified and scoring continues at the field recognition task. If form identification is incorrect, the scoring of outcomes from any subsequent tasks is discontinued. The form is tallied as incorrectly identified and the fields and characters on the form are tallied as missing. These interactions are illustrated in Figure 7.

Given the application description, character-field recognition is dependent on the outcomes from character recognition so that character recognition analysis is conducted first. The inferring of character-field recognition responses from the character recognition task is represented in the scoring flow shown in Figure 8 with the character recognition task nested within the field recognition task. For each field on a correctly identified form, the hypothesized characters generated by the recognition system when reading the field are reconciled with the reference string describing what was entered in the field. This is done through the use of the dynamic string alignment algorithm discussed in Appendix A. The alignments produced are used to tally the number of correct, incorrect (substituted and inserted), and missing characters. These character recognition task results are then used to infer character-field recognition task responses.

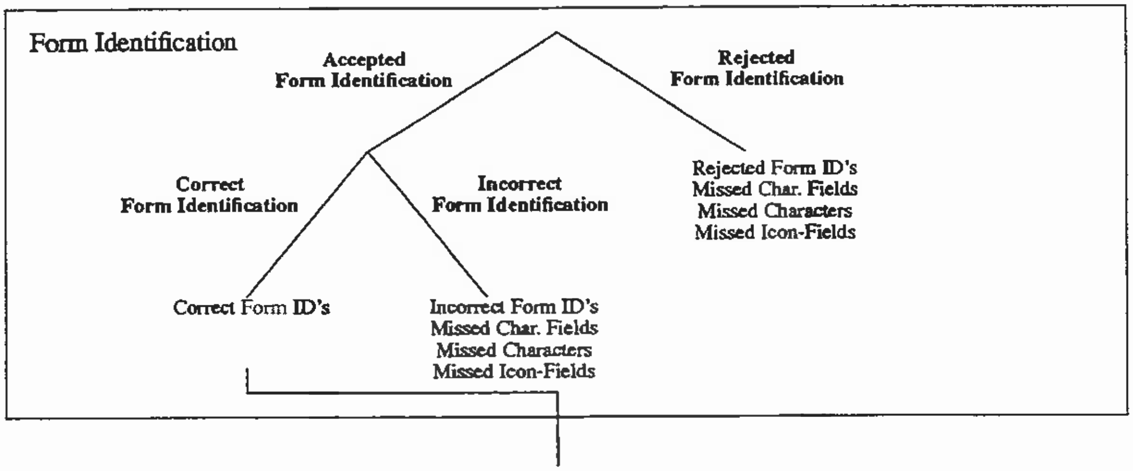


Figure 7: Scoring of form identifications based on a system's ability to reject results.

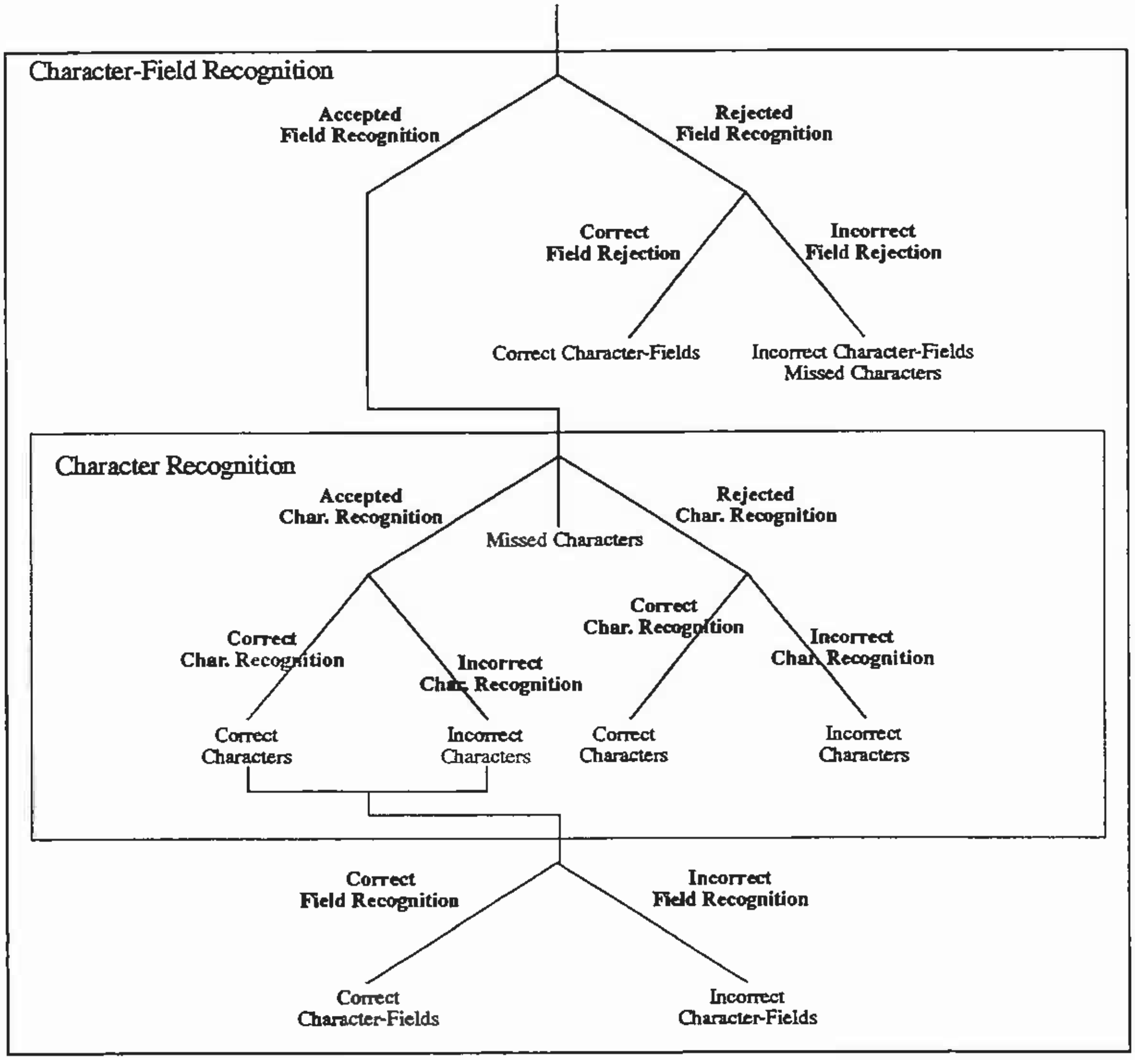


Figure 8: Scoring of character-field and character recognitions based on a system's ability to reject results.

The application description included recognition system rejection decisions at the form identification task, field recognition task, and character recognition task. If the hypothesized identification of a form is rejected, then all the fields and characters on the form are tallied as being missed. Only those fields belonging to forms whose identification is accepted continue to be analyzed at the field recognition task. Upon acceptance of a form's identification, the recognition system may decide to reject the field due to poor recognition of the contents within the field. For example, the confidence values of the individual characters recognized in the field may be too low to place confidence in the successful reading of the entire field. Alternatively, the recognition application may require the recognition and matching of a word in a given field to a predetermined list of words in a dictionary, and the characters recognized in the field may not match any of the words. In both examples, the recognition system determines that the entire field should be rejected.

In the character recognition task, any classification resulting from the recognition of a segmented image may be rejected. It is desirable for a system to reject classifications associated with incorrectly segmented images such as split or merged characters and images of noise. These segmentation errors result in characters being missed (deletion errors) and in erroneous additional classifications being made (insertion errors). It is also desirable to reject incorrect classifications associated with correctly segmented character images. These represent the substitution errors in the system. Unfortunately, rejection mechanisms are not perfect, so that occasionally, correctly classified character images are also rejected. Having described the various instances of character level rejections, a character-field is considered correctly recognized only if every character in the field's reference string has been correctly classified with no characters missed and there are no additional (inserted) classifications remaining after rejection.

The recognition system is also required to detect the presence or absence of non-character information in icon-fields. In this paper, icon-fields include information such as check marks, signatures, and mark-sensed fields. For the application studied in this section, only fields containing check marks and signatures exist on the forms. The scoring flow for the recognition of icon-fields is illustrated in Figure 9. The analysis is simplified because the recognition system only detects the presence or absence of icon-field information and does not have to further recognize the contents of the icon-field.

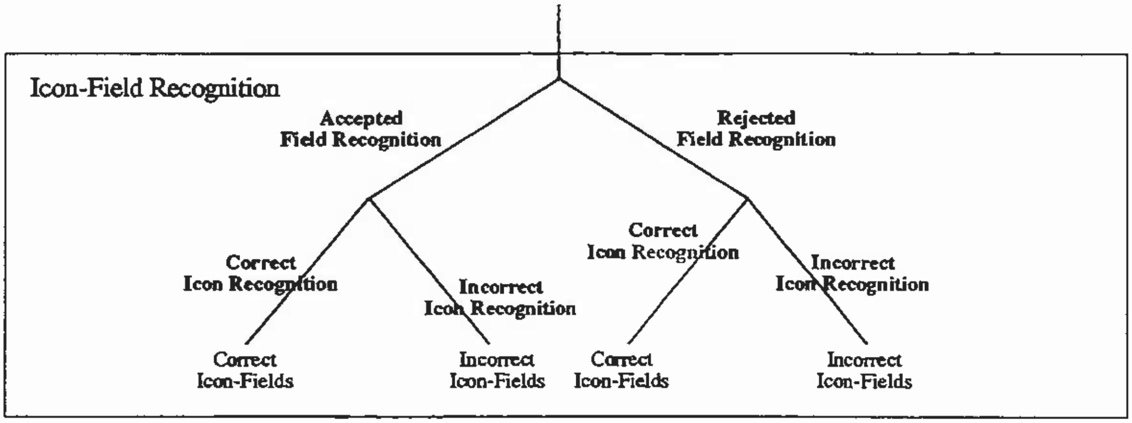


Figure 9: Scoring of icon-fields based on a system's ability to reject results.

4.4 Scoring Accumulators

Given the scoring flows in Figure 7, Figure 8, and Figure 9, scoring variables are defined using an adapted version of the variable nomenclature described in Section 3.4. These variables are used to accumulate recognition system performance statistics within the scoring flows. For this application, variable names may contain one or more letters representing a type of recognition system decision and/or scoring outcome. These letters include recognition system decisions to accept (A) or reject (R) system results. They also include scoring outcomes that determine if the recognition system's decisions were correct (C) or incorrect (I) or whether information was missed (M) by the recognition system. Performance statistics are accumulated at the form, character-field, icon-field, and character levels and are represented as the variable subscripts *form*, *chrfld*, *icofld*, and *char* respectively. Notice that the field recognition task contains two separate sets of accumulators, one for character-fields and one for icon-fields. The form processing task contributing to a particular statistical accumulator is denoted by the variable's superscript. Statistics accumulated for forms identification are denoted as *frmid*, field recognition as *fldrec*, and character recognition as *chrrec*. Notice that the field identification task is not represented by a superscript because the task is not embodied by the target application description and is not included in the scoring flows above.

Using this adapted nomenclature, the variable accumulators used for this application are shown in Figure 10, Figure 11, and Figure 12. For example, the variable RC_{char}^{chrrec} represents the total number of correctly recognized characters rejected by the recognition system. Likewise, the variable representing the total number of character-fields missed due to the recognition system accepting incorrectly assigned form identifications is AM_{chrfld}^{frmid}. These scoring accumulators are summarized in the table shown in Figure 13. Several other accumulators are also required for scoring this application. They include the total number of forms processed $total_{form}$, the total number of character-fields processed $total_{chrfld}$, the total number of icon-fields processed $total_{icofld}$, and the total number of reference characters on the forms processed $total_{refchr}$.

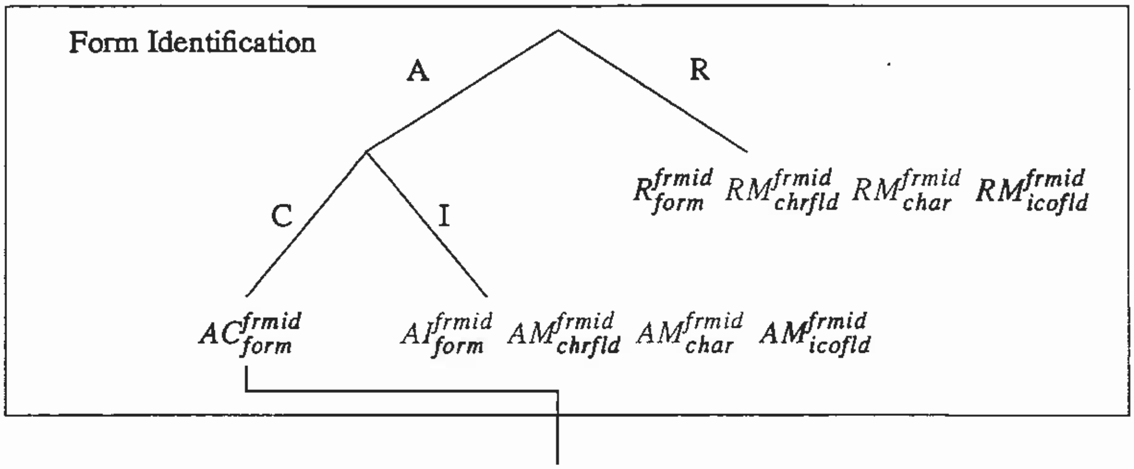


Figure 10: Scoring accumulators for form identifications.

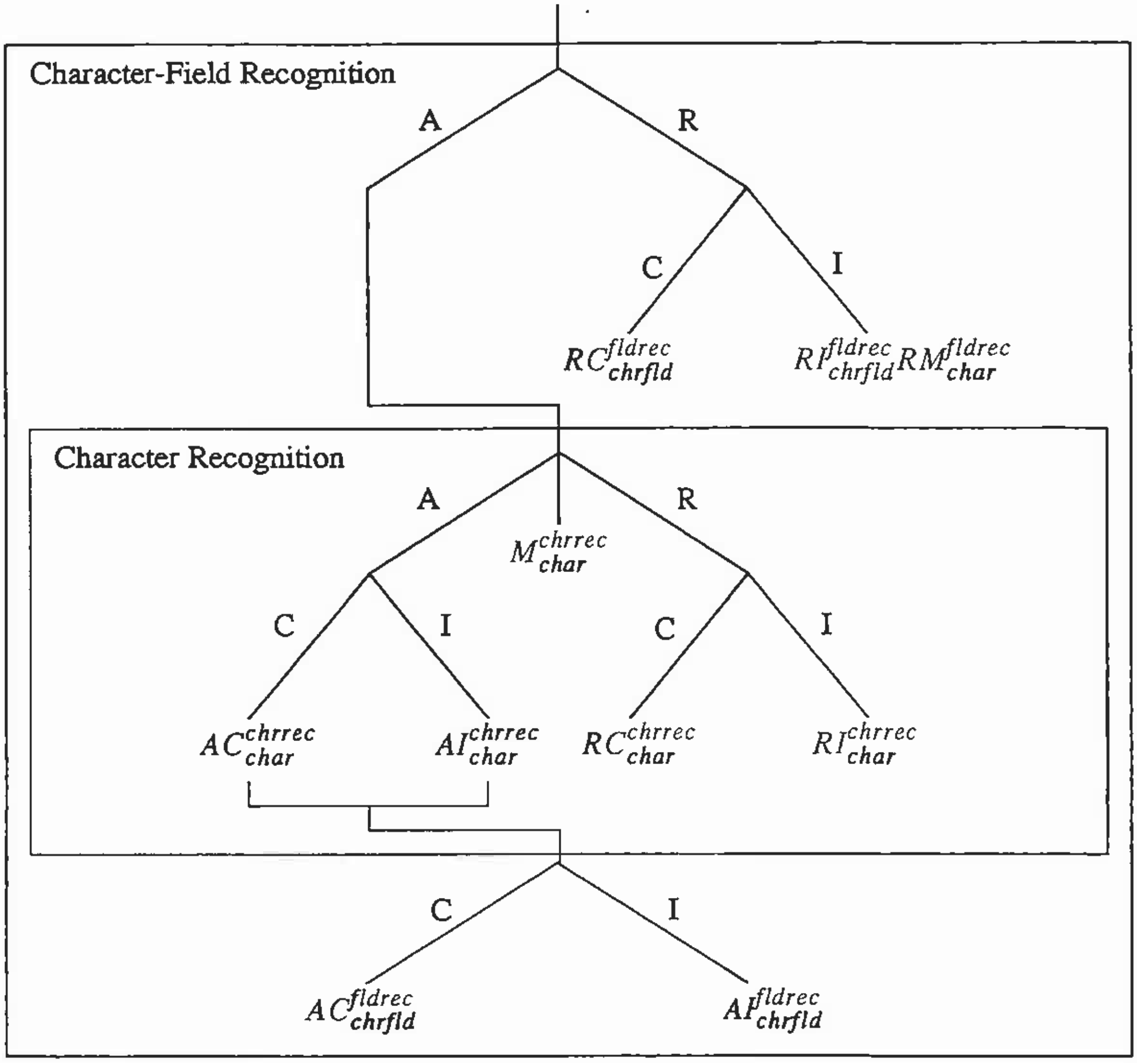


Figure 11: Scoring accumulators for character-field recognitions and character recognitions.

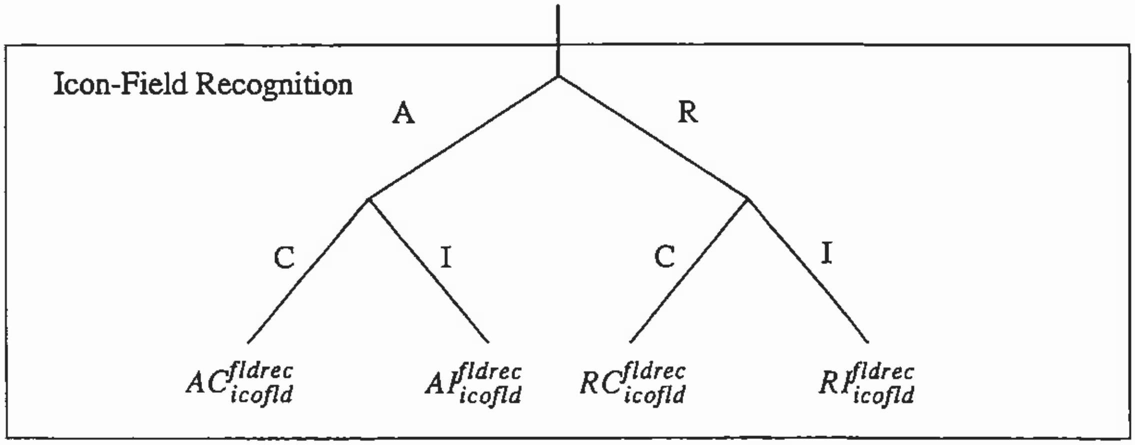


Figure 12: Scoring accumulators for icon-field recognitions.

Totals

1.	$total_{form}$	forms process
2.	$total_{chrfld}$	character-fields processed
3.	$total_{icofld}$	icon-fields processed
4.	$total_{refchr}$	reference characters on all forms processed

Form Identification

5.	AC^{frmid}_{form}	accepted correctly identified forms
6.	AI^{frmid}_{form}	accepted incorrectly identified forms
7.	R^{frmid}_{form}	rejected form identifications

Character-Fields

8.	AM^{frmid}_{chrfld}	missed character-fields due to accepted incorrectly identified forms
9.	AM^{frmid}_{char}	missed characters due to accepted incorrectly identified forms
10.	RM^{frmid}_{chrfld}	missed character-fields due to rejected form identifications
11.	RM^{frmid}_{char}	missed characters due to rejected form identifications

Icon-Fields

12.	AM^{frmid}_{icofld}	accepted missed icon-fields due to accepted incorrectly identified forms
13.	RM^{frmid}_{icofld}	missed icon-fields due to rejected form identifications

Field Recognition

Character-Fields

14.	AC^{fldrec}_{chrfld}	accepted correctly recognized character-fields
15.	AI^{fldrec}_{chrfld}	accepted incorrectly recognized character-fields
16.	RC^{fldrec}_{chrfld}	correctly rejected character-field recognitions
17.	RI^{fldrec}_{chrfld}	incorrectly rejected character-field recognitions
18.	RM^{fldrec}_{char}	missing characters due to incorrectly rejected char. field recognitions

Field Recognition Cont'd

Icon-Fields

19.	AC_{icofld}^{fldrec}	accepted correctly recognized icon-fields
20.	AI_{icofld}^{fldrec}	accepted incorrectly recognized icon-fields
21.	RC_{icofld}^{fldrec}	rejected correctly recognized icon-fields
22.	RI_{icofld}^{fldrec}	rejected incorrectly recognized icon-fields

Character Recognition

23.	AC_{char}^{chrrec}	accepted correctly recognized characters
24.	AI_{char}^{chrrec}	accepted incorrectly recognized characters
25.	M_{char}^{chrrec}	missing characters
26.	RC_{char}^{chrrec}	rejected correctly recognized characters
27.	RI_{char}^{chrrec}	rejected incorrectly recognized characters

Figure 13: Table of scoring accumulators and their definitions.

4.5 Performance Measures

This section presents various performance measures that can be computed from the scoring accumulators defined in Figure 13. The following equations are examples of various recognition system performance measurements and are summarized in the table shown in Figure 14.

4.5.1 Form-Based Performance Measures

$$FORM1 = \frac{AC^{frmid}_{form}}{total_{form}} \qquad (1)$$

$$FORM2 = \frac{AI^{frmid}_{form} + R^{frmid}_{form}}{total_{form}} \qquad (2)$$

$$FORM3 = \frac{AC^{frmid}_{form}}{AC^{frmid}_{form} + AI^{frmid}_{form}} \qquad (3)$$

$$FORM4 = \frac{AI^{frmid}_{form}}{AC^{frmid}_{form} + AI^{frmid}_{form}} \qquad (4)$$

$$FORM5 = \frac{R^{frmid}_{form}}{total_{form}} \qquad (5)$$

4.5.2 Character-Field Based Performance Measures

$$CHRFLD1 = \frac{AC^{fldrec}_{chrfld}}{total_{chrfld}} \qquad (6)$$

$$CHRFLD2 = \frac{AC^{fldrec}_{chrfld}}{AC^{fldrec}_{chrfld} + AI^{fldrec}_{chrfld}} \tag{7}$$

$$CHRFLD3 = \frac{RM^{frmid}_{chrfld}}{total_{chrfld}} \tag{8}$$

$$CHRFLD4 = \frac{AM^{frmid}_{chrfld}}{total_{chrfld}} \tag{9}$$

4.5.3 Icon-Field Based Performance Measures

$$ICOFLD1 = \frac{AC^{fldrec}_{icofld}}{total_{icofld}} \tag{10}$$

$$ICOFLD2 = \frac{AC^{fldrec}_{icofld}}{AC^{fldrec}_{icofld} + AI^{fldrec}_{icofld} + RC^{fldrec}_{icofld} + RI^{fldrec}_{icofld}} \tag{11}$$

$$ICOFLD3 = \frac{RM^{frmid}_{icofld}}{total_{icofld}} \tag{12}$$

$$ICOFLD4 = \frac{AM^{frmid}_{icofld}}{total_{icofld}} \tag{13}$$

4.5.4 Combined Field-Based Performance Measures

$$FIELD1 = \frac{AC_{chrfld}^{fldrec} + AC_{icofld}^{fldrec}}{total_{chrfld} + total_{icofld}} \tag{14}$$

$$FIELD2 = \frac{AC_{chrfld}^{fldrec} + AC_{icofld}^{fldrec}}{AC_{chrfld}^{fldrec} + AI_{chrfld}^{fldrec} + RC_{chrfld}^{fldrec} + RI_{icofld}^{fldrec} + AI_{icofld}^{fldrec} + RC_{icofld}^{fldrec} + RI_{icofld}^{fldrec}} \tag{15}$$

$$FIELD3 = \frac{RM_{chrfld}^{frmid} + RM_{icofld}^{frmid}}{total_{chrfld} + total_{icofld}} \tag{16}$$

$$FIELD4 = \frac{AM_{chrfld}^{frmid} + AM_{icofld}^{frmid}}{total_{chrfld} + total_{icofld}} \tag{17}$$

4.5.5 Character-Based Performance Measures

$$CHAR1 = \frac{AC_{char}^{chrrec} + RC_{char}^{chrrec}}{AC_{char}^{chrrec} + AI_{char}^{chrrec} + RC_{char}^{chrrec} + RI_{char}^{chrrec} + RM_{char}^{frmid} + RM_{char}^{fldrec}} \tag{18}$$

$$CHAR2 = \frac{AC_{char}^{chrrec} + RC_{char}^{chrrec}}{AC_{char}^{chrrec} + AI_{char}^{chrrec} + RC_{char}^{chrrec} + RI_{char}^{chrrec}} \tag{19}$$

$$CHAR3 = \frac{AC_{char}^{chrrec}}{AC_{char}^{chrrec} + AI_{char}^{chrrec}} \tag{20}$$

$$CHAR4 = \frac{RC_{char}^{chrrec} + RI_{char}^{chrrec}}{total_{refchr}} \tag{21}$$

$$CHAR5 = \frac{RC_{char}^{chrrec} + RI_{char}^{chrrec}}{AC_{char}^{chrrec} + AI_{char}^{chrrec} + RC_{char}^{chrrec} + RI_{char}^{chrrec}} \tag{22}$$

$$CHAR6 = \frac{RC_{char}^{chrrec}}{AC_{char}^{chrrec} + RC_{char}^{chrrec}} \tag{23}$$

$$CHAR7 = \frac{RM_{char}^{frmid}}{total_{refchr}} \tag{24}$$

$$CHAR8 = \frac{AC_{char}^{chrrec}}{total_{refchr}} \tag{25}$$

$$CHAR9 = \frac{AC_{char}^{chrrec}}{AC_{char}^{chrrec} + AI_{char}^{chrrec} + RC_{char}^{chrrec} + RI_{char}^{chrrec}} \tag{26}$$

$$CHAR10 = \frac{AM_{char}^{frmid}}{total_{refchr}} \tag{27}$$

Form-Based		
(1)	FORM1	fraction of all forms accepted and correctly identified
(2)	FORM2	fraction of all forms not accepted and correctly identified
(3)	FORM3	fraction of accepted forms correctly identified
(4)	FORM4	fraction of accepted forms incorrectly identified
(5)	FORM5	fraction of all form identifications rejected
Character-Field Based		
(6)	CHRFLD1	fraction of all character-fields accepted and correctly recognized
(7)	CHRFLD2	fraction of accepted character-fields correctly recognized
(8)	CHRFLD3	fraction of all character-fields missed due to rejected form identifications
(9)	CHRFLD4	fraction of all character-fields missed due to accepted incorrect form identifications
Icon-Field Based		
(10)	ICOFLD1	fraction of all accepted and correctly recognized icon-fields
(11)	ICOFLD2	fraction of accepted and correctly recognized icon-fields from all accepted icon-field identifications
(12)	ICOFLD3	fraction of all icon-fields missed due to rejected form identifications
(13)	ICOFLD4	fraction of all icon-fields missed due to accepted incorrect form identifications
Combined Field-Based		
(14)	FIELD1	fraction of all fields accepted and correctly recognized
(15)	FIELD2	fraction of accepted and correctly identified fields correctly recognized
(16)	FIELD3	fraction of all fields missed due to rejected form identifications
(17)	FIELD4	fraction of all fields missed due to accepted incorrect form identifications
Character-Based		
(18)	CHAR1	fraction of correctly recognized characters including characters missed due to rejection
(19)	CHAR2	fraction of correctly recognized characters from all accepted field recognitions
(20)	CHAR3	fraction of accepted correctly recognized characters
(21)	CHAR4	fraction of all character recognitions rejected
(22)	CHAR5	fraction of characters rejected from all accepted field recognitions
(23)	CHAR6	fraction of correctly recognized characters rejected
(24)	CHAR7	fraction of characters missed due to rejected form identifications
(25)	CHAR8	fraction of all accepted and correctly recognized characters
(26)	CHAR9	fraction of accepted and correctly recognized characters from all accepted field recognitions
(27)	CHAR10	fraction of all characters missed due to accepted incorrect field identifications

Figure 14: Table of recognition system performance measures and their definitions.

5. Conclusions

A standard method for measuring the recognition performance of automated form processing systems has been presented. The method requires the specification of a target OCR application, the identification of relevant form processing tasks, the definition of interactions between tasks, the development of a scoring flow, and the definition of scoring accumulators and performance measures. A referenced image database is required that contains both images of typical forms and reference system responses for each of the tasks to be performed on each of the forms. The imaged forms in the database are presented to the recognition system and the system's hypothesized responses are recorded for each of the tasks specified in the application. The hypothesized responses are then processed according to the scoring flow and scoring accumulators are tallied. Once all hypothesized responses are scored, the scoring accumulators are input into performance measures for analysis. This paper presented the general concepts to form-based scoring and demonstrated how they can be applied to a specific application. The methods contained in this paper can be tailored to a wide variety of OCR applications and recognition systems.

6. References

[1] M. D. Garris and S. A. Janet. NIST Scoring Package User's Guide Release 1.0. Technical Report NISTIR 4950, National Institute of Standards and Technology, October 1992.

[2] R. A. Wilkinson, et al. The first Census optical character recognition system conference. Technical Report NISTIR 4912, National Institute of Standards and Technology, July 1992.

[3] M. D. Garris, et al. Massively parallel implementation of character recognition systems. In *Conference on Character Recognition and Digitizer Technologies*, volume 1661, pages 269-280, San Jose California, February 1992. SPIE.

[4] C. L. Wilson and M. D. Garris. Handprinted character database. Technical Report Special Database 1, HWDB, National Institute of Standards and Technology, April 1990.

[5] D. L. Dimmick, M. D. Garris, and C. L. Wilson. Structured Forms Database, Technical Report Special Database 2, SFRS, National Institute of Standards and Technology, December 1991.

[6] D. L. Dimmick and M. D. Garris. Structured Forms Reference Set 2, Technical Report Special Database 6, SFRS2, National Institute of Standards and Technology, September 1992.

[7] M. D. Garris and R. A. Wilkinson. Handwritten segmented characters database. Technical Report Special Database 3, HWSC, National Institute of Standards and Technology, February 1992.

[8] R. A. Wilkinson. Handprinted segmented characters database. Technical Report Test Database 1, TST1, National Institute of Standards and Technology, April 1992.

[9] H. P. Graf, C. Nohl, and J. Ben. Image segmentation with networks of variable scale. J. Moody, S. Hanson, and R. Lippmann, editors, *Advances in Neural Information Processing Systems*, volume IV, pages 480-487. Morgan Kaufmann, Denver, December 1991.

[10] M. D. Garris and C. L. Wilson. A neural approach to concurrent character segmentation and recognition. In *Southcon 92 Conference Record*, pages 154-159, Orlando, March 1992. IEEE.

[11] G. L. Martin. Centered-object integrated segmentation and recognition for visual character recognition. J. Moody, S. Hanson, and R. Lippmann, editors, *Advances in Neural Information Processing Systems*, volume IV, pages 504-511. Morgan Kaufmann, Denver, December 1991.

[12] J. D. Keeler and D. E. Rumelhart. Self-organizing segmentation and recognition neural network. J. Moody, S. Hanson, and R. Lippmann, editors, *Advances in Neural Information Processing Systems*, volume IV, pages 496-503. Morgan Kaufmann, Denver, December 1991.

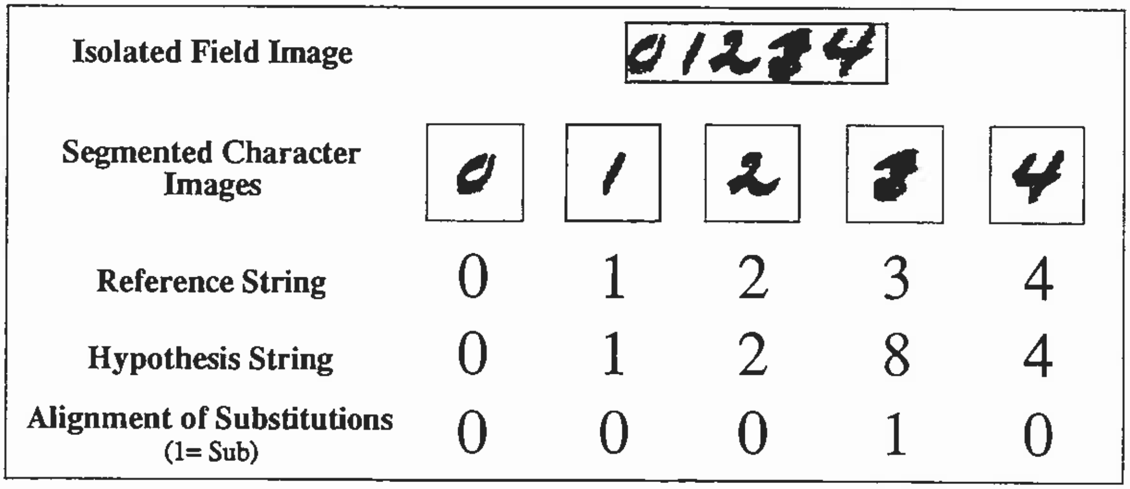
[13] H. G. Zwakenberg. Inexact Alphanumeric Comparison. *The C Users Journal*, pages 127-131. May 1991.

Appendix A: Dynamic String Alignment

The importance of automating the performance assessment of large scale character recognition systems was emphasized in Section 2. The automation of performance assessment methods can be realized through the use of a dynamic string alignment algorithm. This algorithm is responsible for determining how errors occurring in the character recognition task should be assessed. The alignment algorithm reconciles the reference string (what was entered in a field) with the hypothesized string generated by the recognition system. String alignment concepts are discussed in this appendix.

Several different examples are presented in order to demonstrate how string alignments can be used to automatically assess the performance of character recognition systems. The dynamic string alignment algorithm used in the NIST Scoring Package has been adapted from the Levenstein Distance algorithm.[13] This algorithm uses dynamic programming to find the minimum distance between two strings given penalties for character substitutions, deletions and insertions. The algorithm was modified to return the information needed to construct aligned reference and hypothesis strings.

A familiar system error is a substitution error in which the recognition system assigns an incorrect classification to a segmented character image. Figure 15 displays an alignment produced by the Scoring Package of a substitution error caused by an ambiguous character, a '3' classified as an '8'. The hand-printed '3' is malformed so that it really does look ambiguously like an '8' when read by a human. The top image in the figure is an isolated field containing the five hand-printed digits '0', '1', '2', '3', and '4'. The second line of images are the result of segmenting the isolated field into separate images, one character per image. The third line in the figure lists the reference string of what truly was printed in the field. The fourth line lists the hypothesis string corresponding to the assigned classifications generated by the recognition system. The last line in the figure marks the substitution errors identified by the Scoring Package with a '1' representing a substitution error made by the recognition system. As shown in the figure, the segmented character image containing the malformed '3' is classified by the recognition system as an '8' and is identified as a substitution error by the Scoring Package by reconciling the hypothesis string with the reference string.

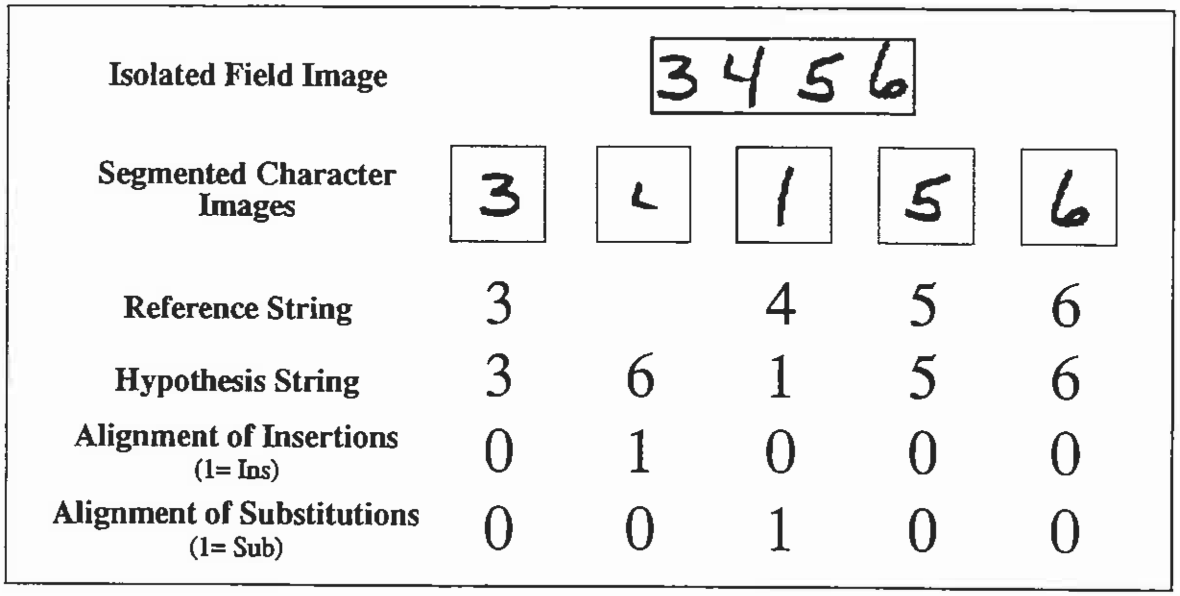


Isolated Field Image					
Segmented Character Images					
Reference String	0	1	2	3	4
Hypothesis String	0	1	2	8	4
Alignment of Substitutions (1= Sub)	0	0	0	1	0

Figure 15: Scoring Package alignment of a substitution error caused by a malformed character.

Another source of character recognition errors comes from incorrectly segmented character images. Most character classifiers are designed to recognize characters one character image at a time. With unconstrained hand-print, characters frequently touch or overlap making the clean separation of characters difficult. Unfortunately, characters are not always segmented correctly. This results in isolated images containing partial characters, multiple characters, and noise. These segmented images are in turn passed to the system's character classifier. Typical segmentation failures result in the insertion of character-like images into, and the deletion of legitimate character images from, the recognition system. This is demonstrated in the examples shown in Figure 16 and Figure 17.

Figure 16 shows an example alignment produced by the Scoring Package of an insertion error caused by a segmentation failure. The top image is an isolated field containing the four hand-printed digits '3', '4', '5', and '6'. The second line of images is the result of segmenting the isolated field into separate images, which are assumed to be one character per image. Notice the '4' has been incorrectly separated into two pieces resulting in two isolated images with two strokes forming a right angle in the left image and a vertical stroke in the right image. This is an example of a segmentation failure, the splitting of a character into multiple images. The third line in the figure lists what was printed in the field. The fourth line lists the hypothesis string corresponding to the assigned classifications generated by the recognition system.



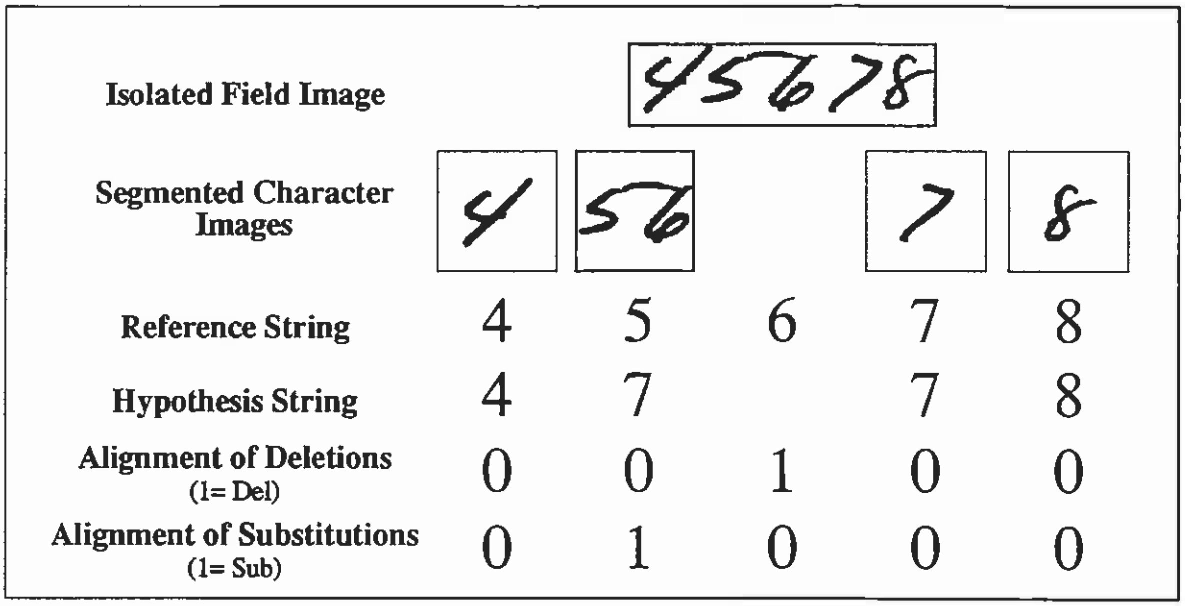
Reference String	3		4	5	6
Hypothesis String	3	6	1	5	6
Alignment of Insertions (1= Ins)	0	1	0	0	0
Alignment of Substitutions (1= Sub)	0	0	1	0	0

Figure 16: Scoring Package alignment of an insertion error caused by a segmentation failure and resulting in a substitution error.

Due to the segmentation failure, the character classifier in the recognition system is presented the two pieces of the '4' rather than one complete character. The result can be seen in the hypothesis string where the first piece of the '4' is classified as a '6' and the second piece of the '4' is classified as a '1'. The fifth line in the figure marks the insertion error identified by the Scoring Package with a '1' representing the inserted classification of a '6'. Often, a single segmentation failure introduces multiple errors into the system. This can be seen by the last line in the figure which marks a substitution error at the position of the second piece of the '4', the vertical stroke. If segmentation failures go undetected, then the character classifier assumes the resulting isolated images are correct and the classifier will assign a classification to each isolated image it is permitted to see. By reconciling the reference string to the hypothesis string, the Scoring Package labeled the second piece of the '4' as a substitution error, knowing that a '4' was truly printed in the field.

Figure 17 shows an example alignment produced by the Scoring Package of a deletion error caused by a segmentation failure. The top image is an isolated field containing the five digits '4', '5', '6', '7', and '8'. The second line of images is the result of segmenting the isolated field into separate images, which are assumed to be one character per image. Notice the '5' and '6' have been merged into a single isolated image. This is another example of a segmentation failure, the merging of multiple characters into a single segmented image. The third line in the figure lists the reference string of what truly was printed in the field. The fourth line lists the hypothesis string corresponding to the assigned classifications generated by the recognition system.

Due to the segmentation failure, the character classifier in the recognition system is presented a single image containing two characters rather than two separate images each containing one character. This is another example of how, if a segmentation failure goes undetected, the character classifier will assign a classification to each isolated image it is permitted to see.The result can be seen in the hypothesis string where the number of assigned classifications is one less than the length of the reference string, and the merged image containing the '5' and '6' is classified as a '7'. The fifth line in the figure marks the deletion error identified by the Scoring Package with a '1' representing the position of the deleted character. The last line in the figure marks the substitution error resulting from the merged character image being incorrectly classified. By reconciling the reference string to the hypothesis string, the Scoring Package located the position of the deleted character and labeled the classification assigned to the merged character image as a substitution error.



Isolated Field Image					
Segmented Character Images					
Reference String	4	5	6	7	8
Hypothesis String	4	7		7	8
Alignment of Deletions (1= Del)	0	0	1	0	0
Alignment of Substitutions (1= Sub)	0	1	0	0	0

Figure 17: Scoring Package alignment of a deletion error caused by a segmentation failure and resulting in a substitution error.

The examples in Figure 16 and Figure 17 demonstrate how system errors have a cascading effect, resulting in multiple errors being introduced into a single hypothesis string. The alignment examples shown are, by design, easy to understand and are easily derived. In practice, multiple errors frequently occur in a single hypothesis string resulting in many different possible alignments. The Scoring Package analyzes each candidate alignment and chooses the one that assesses the least amount of penalty. The Scoring Package does this in a consistent and logical way so that, when given the same hypothesis string and reference string, the Scoring Package will always generate the same alignment. As multiple errors are introduced into the hypothesis string, it becomes increasingly more difficult for the Scoring Package to unambiguously distinguish insertion errors from substitutions errors. This distinction often requires human inspection which would compromise the degree to which the Scoring Package is automated. Therefore, the Scoring Package does not distinguish substitution errors from insertion errors and lumps them together into a single category called false positives.