A General Approach to Quality Evaluation of Document Segmentation Results

Michael Thulke[1], Volker Märgner[1], and Andreas Dengel[2]

[1] Institute for Communications Technology, Braunschweig Technical University
Schleinitzstraße 22, D-38092 Braunschweig, Germany
maergner@ifn.ing.tu-bs.de

[2] German Research Center for Artificial Intelligence (DFKI)
Erwin-Schrödinger-Straße, D-67663 Kaiserslautern, Germany
dengel@dfki.uni-kl.de

Abstract. In order to increase the performance of document analysis systems a detailed quality evaluation of the achieved results is required. By focussing on segmentation algorithms, we point out that the results produced by the module under consideration should be evaluated directly; we will show that the text-based evaluation method which is often used in the document analysis domain does not accomplish the purpose of a detailed quality evaluation. Therefore, we propose a general evaluation approach for the comparison of segmentation results which is based on the segments directly. This approach is able to handle both algorithms that produce complete segmentations (partition) and algorithms that only extract objects of interest (extraction). Classes of errors are defined in a systematic way, and frequencies for each class can be computed. The evaluation approach is applicable to segmentation or extraction algorithms in a wide range. We have chosen the character segmentation task as an example in order to demonstrate the applicability of our evaluation approach, and we suggest to apply our approach to other segmentation tasks.

1 Introduction

Quality evaluation (or benchmarking) is gaining in importance because of several reasons: the increasing quality of document analysis systems during the last few years has made it more and more difficult to achieve a further increase in quality.[1,2] Moreover, the complexity of the systems has increased; the effect is that modifications within one module — even if just one parameter is modified — may often lead to an unpredictable behaviour towards other modules. The increasing amount of ready-to-use algorithms and the exploitation of new application fields for document analysis are making it rather difficult to find a suitable configuration. In conclusion, both, a detailed qualitative and a detailed quantitative failure analysis are needed for further improvement.

When making an evaluation, there may exist two different objectives:

S.-W. Lee and Y. Nakano (Eds.): DAS'98, LNCS 1655, pp. 43–57, 1999.

- Benchmarking for the user. In this case, only the final results (e. g. the ASCII text from an OCR system or a set of categories from a document categorization system) are of interest – there is no motivation to look at internal details.
- Benchmarking for the system developer. Here, a detailed failure analysis is necessary. This requires to focus not only on final results, but on intermediate results, i.e. on the output of the specific module. The module's output must be accessible and needs to be compared with the corresponding ground truth. Our focus is lying on this objective.

There are several possibilities to evaluate a module (we will discuss them later). The way we do it is empirical, not analytical. That means we are using the results which the module produces for evaluation. The module under consideration is a black box — we do not want to perform an algorithmic analysis. The module's results are compared with ideal results (ground truth).

In this paper, the modules under consideration are segmentation modules of document analysis systems. The module's output as well as the ground truth are segments. That means, they are of a geometrical, not of a symbolic kind. Thus, the evaluation is made on the basis of segments.

The paper is tructured as follows: in the next section we examine the problem of segmentation in document analysis systems. Section 3 discusses other evaluation approaches which can be used to benchmark segmentation modules. Section 4 shows the disadvantages of text-based evaluation in the case of focussing on a detailed failure analysis. In section 5 we propose our general approach to the evaluation of segmentation results. This approach was successfully taken to character segmentation. The application to character segmentation is subject of section 6. We finish with a conclusion.

2 Segmentation Modules in Document Analysis Systems

In a document analysis system where image areas have to be segmented several tasks have to be carried out.

Consider the layout analysis task. After deskewing each document page, the layout has to be analysed. This task, also denoted as *zoning*, consists primarily of segmenting the image into text blocks and non-text blocks, and is followed by the determination of a reading order for text blocks. There is a broad spectrum of approaches to the layout analysis problem. Early work was done by Wahl et al.[3] Later a survey paper was written by Haralick.[4]

The extraction of regions-of-interest, e.g. the address block location in a mail-sorting task, can be considered as a special case of the segmentation task. The difference lies only in the fact that, in address block location, it is known that exactly one segment has to be found.

The way from text blocks to isolated characters involves several segmentation steps: the text lines have to be isolated, followed by a segmentation of each line into words and characters. These segmentation tasks have in common that the segmentation primarily goes on in one direction: line segmentation[1] from top to bottom, word and character segmentation from left to right — in contrast to the layout analysis task, where, in general, no total ordering of the segments exists. Nevertheless, in each case the segments are two-dimensional objects. For line, word and character segmentation, there is a number of papers proposing a wide range of solutions. The main emphasis lies on character segmentation, which is not surprising, because this seems to be the most crucial topic. For a survey of character segmentation methods, see e.g. Casey and Lecolinet.[5]

3 Other Contributions to the Segmentation Evaluation Task

There are several approaches to the evaluation of segmentation results or, more generally, to the evaluation of pattern analysis tasks. Figure 1 gives a classification scheme of the different approaches.

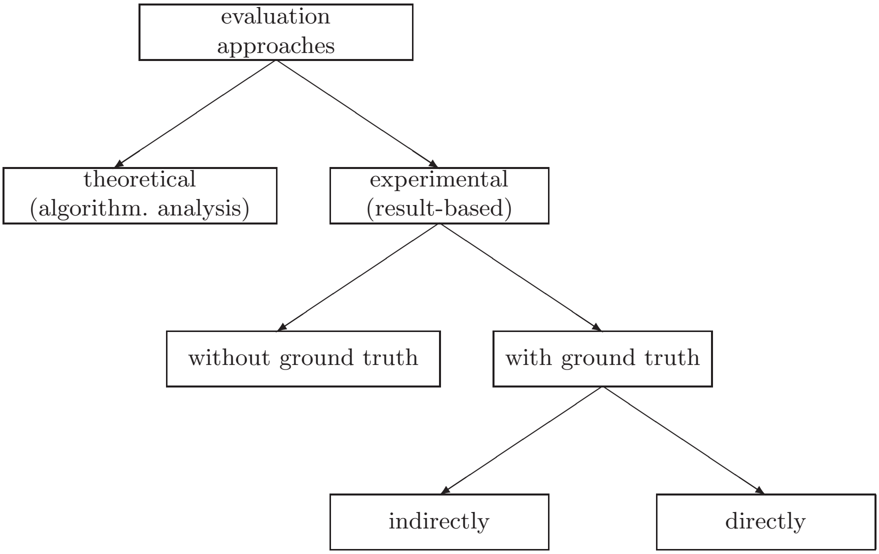


Fig. 1. Classification of evaluation approaches

First we can distinguish between a theoretical analysis and an experimental approach. In the theoretical approach, the algorithm will be analysed in order to derive its behaviour. The input data are assumed to be composed of ideal data

[1] By line segmentation we mean the segmentation *into* lines; in the same way, we say character segmentation when we mean the segmentation *into* characters.

degraded by noise. Input data degradation is propagated analytically throughout the algorithm. E.g., Haralick proposed an approach that uses covariance propagation.[6] —Approaches like this one are mainly used in the domain of low-level computer vision algorithms; edge detection would be an example for this. This approach would hardly be applicable since segmentation modules from the document analysis domain do not belong to the class of analyzable low-level algorithms.

On the contrary, experimental evaluation considers a module that should be characterized as a black box. The evaluation is based on the results only. This approach is much more pragmatic and is independent of the underlying algorithm.

It is possible to implement some *on-line measures* which can be used to evaluate a specific aspect of quality. On-line means that the system itself can calculate the measure of assessing the quality of its results. Ground truth is not necessary. But when making a detailed and comprehensive quality evaluation, this methodology cannot be sufficient.

When using ground truth, quality evaluation means to make a comparison. It depends on the type of data whether the comparison is simple or more complicated. It can easily be seen that the comparison of segmentation results is more complicated than e.g. the comparison of character classification results because segmentation results are *(i)* not isolated like character patterns nor *(ii)* of a symbolical kind.

Not least because of this, there exists the idea that comparison may not be performed on the basis of the direct results, but on the basis of indirect results. That means, a further processing of the results has been done. In this domain, the work of ISRI is worth mentioning. They compare the textual results which a document analysis system generates with the plain ground truth text in order to evaluate the zoning quality of a system.[7] The comparison is done by means of a string-matching algorithm. ISRI conducted an annual test of the accuracy of several commercially available OCR systems.[8] Zoning evaluation was a part of the test. A great advantage of this approach is that ground truth is only needed on a textual level which is easier to create than ground truth on the segment level. But the disadvantage is, that a quality evaluation which reports in detail what has happened cannot be perfomed. This is due to the loss of information that occurs when the segments are converted to a stream of character symbols.

This disadvantage does not exist when an evaluation s made on the basis of direct results, say, of the segment data itself. When this approach is used, the segmentations have to be compared. Several authors have worked on this approach, both from the document analysis community and from the computer vision community, in general. Yanikoglu and Vincent used this approach to evaluate zoning results.[9] Chen has worked on segmentation algorithms and developed a segmentation comparison procedure to evaluate their performance.[10] Hoover et al. developed a procedure for segmentation comparison of range images; [11] but their method of comparing segmentations can be transferred to the document analysis domain without problems.

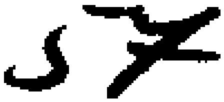
Beside the question how to perform the evaluation there is the task to provide test data. There are two extremes: the collection of real data and, since this is often a very laborious task, the generation of synthetic data. For several years, the document analysis community was using synthetic images including image degradation models. First work on this topic was done by Baird.[12]

It can be stated that hardly any efforts were taken to carry out evaluation on the character segmentation level. We mentioned before that character segmentation is a very crucial topic. Moreover, it was reported that the majority of errors is due to incorrect segmentation rather than incorrect character recognition. This is true for constrained handwriting as well as for printed text.[14,15]

4 Evaluation on the Textual Level and Its Disadvantages

Evaluation on textual results will have several disadvantages in view of assessing the quality of a (character) segmentation module. Consider the character recognition task of a text field that usually consists of several characters. The system's output text string is compared with the ground truth string; usually a string-matching algorithm using the Levenshtein distance, a well-known method based on the dynamic programming principle, is used. When having applied this matching procedure, four different classes (of errors) may occur: *(i)* a character is *correctly* recognized; *(ii)* a character is incorrectly recognized *(substitution)*; *(iii)* a character is incorrectly *inserted*; *(iv)* a character is incorrectly *deleted.* Criticism of the text-based evaluation approach is summarized in the following three points:

1. *The text-based approach is not able to distinguish between segmentation and recognition errors.*
   Interference between segmentation errors and recognition errors is possible. Figure 2 gives an example where an erroneous segmentation has been classified as a substitution error which creates the impression that there has been a recognition error.



ground truth text: 57*; detected text:* 37'
⇒ *errors made: 1 × substitution.*

Fig. 2. Example of confusion regarding recognition and segmentation error

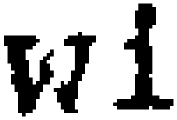
2. *The classes of errors are not suited to the problem.*
   Consider the class *insertion.* Even when the text result has an insertion it is not clear what really caused it. The cause for an insertion effect may be

an erroneous interpretation of a noise-like object as a character image. But an insertion might also be caused by an erroneous splitting operation. When making a text-based evaluation, one cannot decide which specific type of segmentation error has occurred.

3. *The match which is detected cannot be guaranteed to be the correct one.*

   Figure 3 shows a part of a document image, the ground truth text and the detected text. The character 'w' was splitted up into two fragments; these fragments were classified as 'v' and 'i'. The character 'i' was misclassified as a 'l'. The ground truth text 'i' was brought into match with the character 'i' in the output string; this alignment is wrong because the underlying image parts are disjoint.

   The goal of the string-matching algorithm is to find the alignment referring to the minimum transformation cost; but this alignment is only useful for describing the effort that a subsequent system has to perform in order to achieve correct results. Therefore, the alignment only describes the error's effect but not the error itself, a fact which becomes obvious especially when looking at the errors the segmentation module can cause.



ground truth text: `wi`*; detected text:* `vil`
⇒ *errors made: 1 × substitution, 1 × insertion.*

Fig. 3. Example of an incorrect match owing to lack of geometrical knowledge

5 An Approach to the Evaluation of Segmentation Results

In the following section we are presenting a general approach to the evaluation of segmentation algorithms. Our approach is based on geometrical data and can be considered as a framework from which concrete evaluation methods can be derived for use in a specific domain.

In the great majority of cases, a segmentation becomes — to be precise — an extraction (or detection or isolation), because only some objects present in the area to be analyzed are objects of interest and others not. As a consequence, our approach can handle pure segmentation algorithms as well as extraction algorithms.

5.1 Problem Definition

Let I be the set of indices from the area to be segmented, e.g. $I = [0, X) \times [0, Y)$ in a two-dimensional image with dimensions $X \times Y$ (or a part of it)[2]. We now define a *segment* $u_i \subset I$ as an arbitrary non-empty, partial set of I. A *segmentation* is now a partitioning of I into a set of segments, that is *(i)* $I = \bigcup_i u_i$ and *(ii)* $u_i \cap u_j = \emptyset$ for all i, j $(i \neq j)$.

There are two types of segments: segments which denote objects-to-detect and a segment consisting of the remaining area. The first we call hereinafter *segments of interest* and the latter we call *noise segment*. As we will see below, these types of segments will be handled differently.

When making an evaluation, we have a ground truth segmentation $G = G_{obj} \cup \{g_{noise}\}$ with $G_{obj} = \{g_1, \ldots, g_M\}$ and the segmentation detected from the system $S = S_{obj} \cup \{s_{noise}\}$ with $S_{obj} = \{s_1, \ldots, s_N\}$ where g_i resp. s_j denote the segments of interest.

The first step when to compare two segmentations is to achieve relationships between a single segment from G and a single segment from S.

Let $|\cdot|$ denote the number of elements of a set; we define an *overlap function* $G \times S \to \mathcal{N}_0, (g, s) \mapsto |g \cap s|$ which counts the number of pixels common to $g \in G$ and $s \in S$. Given two segmentations G and S, the values of this function can be displayed in a two-dimensional table; there exist the marginal conditions $|g| = \sum_{s \in S} |g \cap s|$, $|s| = \sum_{g \in G} |g \cap s|$ and $|I| = \sum_{g \in G} \sum_{s \in S} |g \cap s|$. The following evaluation methods are based on this overlap function.

5.2 Elementary Classes of Errors

When the errors where a specific segment is participated should be classified, the following elementary classes of errors can be found intuitively:
(i) Several segments were merged together (hereinafter called *merge*);
(ii) a segment was split in several fragments (*split*);
(iii) a segment was completely ignored (*miss*);
(iv) a segment was detected where no segment should be (*false*);
(v) parts of a segment are missing (*partial miss*);
(vi) noise was added to a segment (*partial false*).
The overlapping function described above may be used to derive these types of errors. According to the classes of errors introduced above, we define the following sets of segments:

(i) For the class *merge*: A detected segment belongs to the class *merge* if and only if it overlaps with at least two ground truth segments-of-interest. Or, more formally:

$$S_{Merge} := \{s \in S_{obj} | \bigvee_{g^{(1)}, g^{(2)} \in G_{obj}, g^{(1)} \neq g^{(2)}} |g^{(1)} \cap s| > 0 \wedge |g^{(2)} \cap s| > 0\}.$$

[2] Note that we use only the indices of the pixels, not the pixel values themselves.

Having defined this, one can define the set of the *ground truth* segments which participated:

$$G_{Merge} := \{g \in G_{obj} | \bigvee_{s \in S_{Merge}} |g \cap s| > 0\}.$$

Note that one ground truth segment can participate in merge errors from several (detected) segments.

(ii) In the same way, two sets denote the segments which participate on split errors:

$$G_{Split} := \{g \in G_{obj} | \bigvee_{s^{(1)}, s^{(2)} \in S_{obj}, s^{(1)} \neq s^{(2)}} |g \cap s^{(1)}| > 0 \wedge |g \cap s^{(2)}| > 0\},$$

$$S_{Split} := \{s \in S_{obj} | \bigvee_{g \in G_{Split}} |g \cap s| > 0\}.$$

(iii) The set of segments which have been completely missed is defined as

$$G_{Miss} := \{g \in G_{obj} | \neg \bigvee_{s \in S_{obj}} |g \cap s| > 0\}.$$

(iv) Analogously we define

$$S_{False} := \{s \in S_{obj} | \neg \bigvee_{g \in G_{obj}} |g \cap s| > 0\}.$$

The last two classes of errors, *(v)* and *(vi)*, are defined as follows:

$$G_{PartialMiss} := \{g \in G_{obj} \Big| |g \cap s_{noise}| > 0 \wedge \bigvee_{s \in S_{obj}} |g \cap s| > 0\},$$

$$S_{PartialMiss} := \{s \in S_{obj} | \bigvee_{g \in G_{PartialMiss}} |g \cap s| > 0\},$$

$$S_{PartialFalse} := \{s \in S_{obj} \Big| |g_{noise} \cap s| > 0 \wedge \bigvee_{g \in G_{obj}} |g \cap s| > 0\},$$

$$G_{PartialFalse} := \{g \in G_{obj} | \bigvee_{s \in S_{PartialFalse}} |g \cap s| > 0\}.$$

Having defined these classes of errors, it is obvious to use the number of segments belonging to a set as an error metric. For example, with $M_{Split} = |G_{Split}|$ we can use the absolute frequency M_{Split} or the relative frequency M_{Split}/M as an error metric.

The use of these classes of errors gives a good impression how often a specific type of segmentation error occurs. But it is important to note that these classes of errors are not disjoint. For example, a ground truth segment may be a member of the split class and may be a member of the merge class simultaneously. Therefore, a unique classification into disjoint classes is not possible.

There is a second limitation when using these elementary classes of errors. In some applications, there may be an interest which segments are involved in a concrete segmentation error (this will be demonstrated in section 6.2). Questions like that cannot be handled by this evaluation approach.

In the next section, we propose an approach which overcome these limitations.

5.3 Generation and Classifications of Regions

We define for two arbitrary segments a and b the relation $a \sim b$ if and only if $|a \cap b| > 0$.[3] So $a \sim b$ means that segments a and b overlap significantly. Since the segments within G do not overlap, a and b cannot be both from G. The same is true for segments within S. Let $a \simeq b$ be an equivalence relation induced by $\sim$. The equivalence relation is transitive; as a consequence, $a \simeq b$ is valid if there exists a sequence of segments $u_1, \ldots, u_n$ with $a \sim u_1 \wedge u_1 \sim u_2 \wedge \ldots \wedge u_n \sim b$. According to the $\simeq$ relation we construct the equivalence classes in $G_{obj} \cup S_{obj}$, hereinafter called *regions*. Each region r contains segments from G_{obj} and/or segments from S_{obj}.

The next step is to check for each region r_k whether it overlaps with g_{noise} (that means, noise has been misclassified as an object or as part of an object) and/or whether it overlaps with s_{noise} (object or part of it has been misclassified as noise). More formally, we define the predicates $G_{noise}(k) :\iff |g_{noise} \cap \bigcup_{u \in r_k} u| > 0$ and $S_{noise}(k) :\iff |s_{noise} \cap \bigcup_{u \in r_k} u| > 0$.

We define — for the sake of completeness — $g_{noise} \cap s_{noise}$ being an additional region if it is non-empty. Thereby we obtain that the set of regions is a partitioning of I.

Classification of Regions

We characterize the quality of the segmentation by the quality of its regions. Therefore, we classify each region.

Let $g_{obj}^{(k)}$ be the number of ground truth segments-of-interest of a region r_k and $s_{obj}^{(k)}$ the number of detected segments-of-interest. The following constraints are a consequence of the region building mechanism:

(i) Since the segmentations G and S cover each the whole area of I, each region must at least contain parts from G and parts from S.

(ii) A region which consists only of noise on the ground truth side does not have the capability to combine detected segments. The same holds for a region which consists only of noise on the detected side.

Furthermore, we group the possible values for $g_{obj}^{(k)}$ and $s_{obj}^{(k)}$ into the categories $\{0, 1, >1\}$. Together with taking the above-mentioned constraints into account, this leads to 19 disjoint classes of errors, as shown in table 1.

[3] Alternatively one can define $|a \cap b| > \delta$ where $\delta \geq 0$ is a small fixed threshold value.

	$g_{obj}^{(k)}$	$G_{noise}(k)$	$s_{obj}^{(k)}$	$S_{noise}(k)$	**class name**
1	0	true	0	true	*noise*
2	0	true	1	false	*false*
3	1	false	0	true	*miss*
4	1	false	1	false	*correct*
5	1	false	1	true	*correct incl. object as noise*
6	1	false	>1	false	*split*
7	1	false	>1	true	*split incl. object as noise*
8	1	true	1	false	*correct incl. noise as object*
9	1	true	1	true	*correct incl. object as noise and noise as object*
10	1	true	>1	false	*split incl. noise as object*
11	1	true	>1	true	*split incl. object as noise and noise as object*
12	>1	false	1	false	*merge*
13	>1	false	1	true	*merge incl. object as noise*
14	>1	false	>1	false	*merge+split*
15	>1	false	>1	true	*merge+split incl. object as noise*
16	>1	true	1	false	*merge incl. noise as object*
17	>1	true	1	true	*merge incl. object as noise and noise as object*
18	>1	true	>1	false	*merge+split incl. noise as object*
19	>1	true	>1	true	*merge+split incl. object as noise and noise as object*

Table 1. Classes of errors

Applying this classification scheme to concrete segmentation results, relative frequencies can be computed for each class. Because there are 17 classes where ground truth segments-of-interest are involved, we are able to compute 17 frequencies of ground truth segments-of-interest w. r. t. the total number of ground truth segments-of-interest. In the same way 17 frequencies of detected segments-of-interest w. r. t. the total number of detected segments-of-interest can be computed. Note that the classification scheme is completely symmetrical regarding ground truth and detected segmentation.

By using these measures, it exists a duality: There are e. g. two measures for the merge class, one from the ground truth side and one from the detected side. In order to avoid this duality, the number of regions belonging to each of the 19 classes can alternatively be used. Using this way, we obtain 19 frequencies. A single quality measure — although it is very problematic to characterize such a complex task by one value — can be obtained by computing a weighted sum over the class frequencies. Figure 4 illustrates the basic evaluation approach.

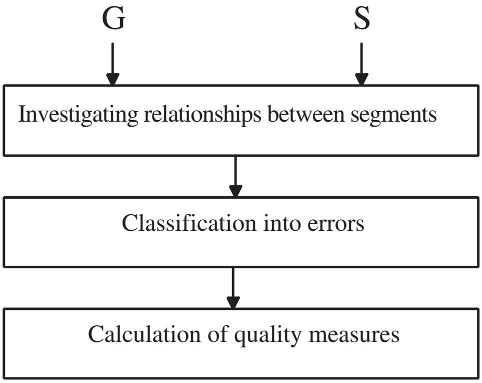

G
S
Investigating relationships between segments
Classification into errors
Calculation of quality measures

Fig. 4. Overview of the basic evaluation approach

5.4 Derivation of Concrete Evaluation Methods

It depends on the individual *evaluation* task whether it is necessary for an evaluation to compute all these values in detail. Moreover it depends on the individual *segmentation* task whether each class of error can occur. Often it is possible to simplify this evaluation approach by uniting some classes of errors.

Consider the example of a noise-free segmentation evaluation. In this case there does not exist either ground truth noise or detected noise. As a consequence, only classes with $\neg G_{noise}(k) \wedge \neg S_{noise}(k)$ can occur. This leads to a simple segmentation evaluation scheme with 4 possible classes when using the region-based approach, as shown in table 2 (the region index k is omitted for purposes of readability).

	g_{obj}	s_{obj}	**class name**
1	1	1	*correct*
2	1	>1	*split*
3	>1	1	*merge*
4	>1	>1	*merge+split*

Table 2. Reduced classification scheme

6 Application to Character Segmentation

In this section we report on the evaluation of a character segmentation module. Our focus lies on the evaluation method itself, neither on the segmentation algorithm nor on the results of the evaluation. We use the segmentation evaluation approach presented in the previous section as a basis; the approach will be extended by some practical considerations (sections 6.2 and 6.3); these extensions are also applicable to other segmentation evaluation tasks.

6.1 Application of Our Evaluation Approach

A character segmentation algorithm usually operates on binary images, where a value of 1, for instance, stands for the foreground, and a value of 0, for the background. Only the segmentation of the foreground is of interest; in particular, the foreground consists of character pixels, speckles, character or field boxes on forms etc. As a consequence, this segmentation algorithm has to distinguish segments-of-interest (characters) from noise (speckles etc.). For evaluating this task, the generation of ground truth is necessary. Therefore, we make the following decisions:

1. The set of indices from the area to be segmented *(I)* contains only the indices of foreground pixels.
2. All non-character foreground objects are declared as noise.
3. Each single character image is exactly one segment-of-interest.

Having made these decisions, the segmentation evaluation framework is applicable.

We used text line images taken from about 400 address blocks printed in various fonts. For each text line, the results of the character segmentation module are organized in a segment hypothesis graph as shown in figure 5.

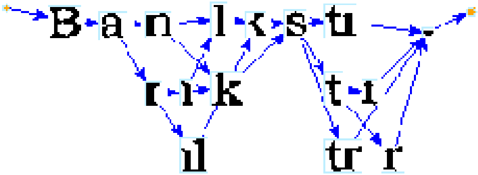


Fig. 5. Segment hypothesis graph

This data structure is very popular when to represent character segmentation results. Each path through the graph represents one possible segmentation. Furthermore, the character classification results (including classifier confidences) may be attached to each segment node.

Ground truth was created half-automatically. In a first step, the best path was selected from the hypothesis graph by using a criterion based on the confidences of the character classification results of each segment. This segmentation was suggested in order to be checked or corrected manually by means of our interactive graphical tool InSegD (*Interactive Segmentation of Documents*).[16]

When doing the evaluation, we generated all possible paths throughout the hypothesis graph and compared them with the ground truth segmentation. For each comparison, we calculated a weighted sum over the class frequencies of the segmentation errors occurred. We chose the path with the best score.

The evaluation tool SEGEVAL implements the approach described in the previous section. Figure 3 gives an exemplary output for the region-based evaluation approach.

```
                                              G                   S              #Regions
-------------------------------------------------------------------------------------------
 1       : 1        [Correct]:     97.973% ( 18657)    98.464% ( 18657)           18657
 1       :(1+noise)          :      0.467% (     89)    0.470% (     89)              89
(1+noise): 1                 :      0.000% (      0)    0.000% (      0)               0
(1+noise):(1+noise)          :      0.000% (      0)    0.000% (      0)               0

 M       : 1        [Merge]  :      0.924% (    176)    0.464% (     88)              88
 M       :(1+noise)          :      0.021% (      4)    0.011% (      2)               2
(M+noise): 1                 :      0.000% (      0)    0.000% (      0)               0
(M+noise):(1+noise)          :      0.000% (      0)    0.000% (      0)               0

 1       : M        [Split]  :      0.152% (     29)    0.306% (     58)              29
 1       :(M+noise)          :      0.005% (      1)    0.011% (      2)               1
(1+noise): M                 :      0.000% (      0)    0.000% (      0)               0
(1+noise):(M+noise)          :      0.000% (      0)    0.000% (      0)               0

 M       : M        [M+S]    :      0.179% (     34)    0.195% (     37)              17
 M       :(M+noise)          :      0.000% (      0)    0.000% (      0)               0
(M+noise): M                 :      0.000% (      0)    0.000% (      0)               0
(M+noise):(M+noise)          :      0.000% (      0)    0.000% (      0)               0

 noise   : 1        [False]  :        -          -      0.079% (     15)              15
 1       : noise    [Miss]   :      0.278% (     53)      -          -                53
-------------------------------------------------------------------------------------------
Total                        :             ( 19043)            ( 18948)
```

Table 3. Exemplary output of the segmentation evaluation tool

6.2 Inclusion of Segment Attributes

Besides the ground truth generation on the segment level, we generated textual reference data, too. So each ground truth segment is attributed by its character class. Therefore, it is possible to select a specific class of segmentation error and to make a further distinction by using the character classes involved. Consider, for example, the class merge. By using the region-based approach, it was possible to detect which sequences of characters were the favourites for a merge. Therefore, it was possible to discover in a quantitative manner that most frequently the letter combinations '`tr`', '`ri`' and '`rf`' lead to a merge.

6.3 Evaluation of Hypothesis Generation

Here, the goal is to assess the quality of the hypothesis graph. The ideal case would be a graph that contains only one path being the correct one. Since the segments are of a geometrical nature, it is possible to count the number of hypotheses a over a specific pixel (x, y) or over a specific ground truth segment g. For example, it could be stated that the average number of hypotheses $(\bar{a}(g_i))$ over uppercase ground truth segments is lower than the average number over lower case segments.

It should be noticed that $\bar{a}(x, y)$ is an *on-line measure* and, therefore, it can be used to detect image areas where the segmentation is unsafe.

7 Conclusion

Recently it has been recognized that the character segmentation task is a very critical one within the list of tasks of a document analysis system. Now as before, too little effort is made to deal with the thorny subject of performance evaluation, especially for the segmentation domain.

First, we pointed out that text-based evaluation of segmentational tasks has several disadvantages, e. g. it only describes the effect of the errors made and not the errors itself. Therefore, it is not sufficient for a detailed error analysis.

Secondly, we proposed a general approach to the evaluation of segmentation or extraction algorithms. In this approach, we introduced elementary classes of segmentation errors that are obvious. Furthermore we developed a segmentation evaluation approach which provides disjoint classes of errors. Since the evaluation is based on the segments directly, it is applicable to segmentation algorithms in a wide range and not restricted to a specific domain.

We demonstrated the applicability of our approach by means of a practical example for the character segmentation task. Besides the application to character segmentation as presented here, we also have applied our evaluation approach to the zoning task. Finally, we want to suggest the use in other segmentation tasks, within and without the document analysis domain.

References

1. T. Pavlidis: *Problems in the Recognition of Poorly Printed Text*, Proc. Symposium on Document Analysis and Information Retrieval, Las Vegas 1992, pp. 162–173 43
2. M. D. Garris: *Method and Evaluation of Character Stroke Preservation on Handprint Recognition*, National Institute of Standards and Technology (NIST) Technical Report NISTIR 5687, July 1995; published in: SPIE, Document Recognition III, pp. 321–332, San Jose, January 1996 43
3. F. M. Wahl, K. Y. Wong, R. G. Casey: *Block Segmentation and Text Extraction in Mixed Text/Image Documents*, Computer Graphics and Image Processing, Vol. 20, 1982, pp. 375–390 44
4. R. M. Haralick: *Document Image Understanding: Geometric and Logical Layout*, CVPR, Seattle, USA, June 1994 44
5. R. G. Casey, E. Lecolinet: *A Survey of Methods and Strategies in Character Segmentation*, IEEE Transactions on Pattern Analysis and Machine Intelligence, July 1996, pp. 690–706 45

6. R. M. Haralick: *Propagating Covariance in Computer Vision*, Workshop on Performance Characteristics of Vision Algorithms, Robin College, Cambridge, UK, April 1996 46
7. J. Kanai, S. V. Rice, T. A. Nartker, G. Nagy: *Automated Evaluation of OCR Zoning*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 17, No. 1, Jan. 1995, pp. 86–90 46
8. S. V. Rice, F. R. Jenkins, T. A. Nartker: *The Fifth Annual Test of OCR Accuracy*, Information Science Research Institute, University of Nevada, Las Vegas, Technical Report ISRI TR-96-01, April 1996 46
9. B. A. Yanikoglu, L. Vincent: *Ground-truthing and Benchmarking Document Page Segmentation*, Proc. 3rd Intern. Conf. on Document Analysis and Recognition (ICDAR), Montréal, Canada, 1995, pp. 601–604 46
10. S. Chen, R. M. Haralick, I. T. Phillips: *Perfect Document Layout Ground Truth Generation Using DVI Files and Simultaneous Word Segmentation From Document Images*, Proc. Fourth Annual Symposium on Document Analysis and Information Retrieval, Las Vegas 1995, pp. 229–248 46
11. A. Hoover, G. Jean-Baptiste, X. Jiang, P. J. Flynn, H. Bunke, D. Goldgof, K. Bowyer, D. Eggert, A. Fitzgibbon, R. Fisher: *An Experimental Comparison of Range Image Segmentation Algorithms*, IEEE Transactions on Pattern Analysis and Machine Intelligence, July 1996, pp. 1–17 46
12. H. S. Baird: *Document Image Defect Models*, in: *Structured Document Image Analysis*, Springer, New York, 1992, pp. 546–556 47
13. M. Thulke: *Use of Geometrical Ground Truth for Quality Evaluation of Document Segmentation Algorithms*, in: W. Förstner (editor): Workshop *Performance Characteristics and Quality of Computer Vision Algorithms*, Braunschweig, Germany, September 1997
14. P. Stubberud, J. Kanai, V. Kalluri: *Adaptive Restoration of Text Images Containing Touching or Broken Characters*, Information Science Research Institute (ISRI) 1995 Annual Research Report, pp. 61–96 47
15. C. L. Wilson, J. Geist, M. D. Garris, R. Chellappa: *Design, Integration and Evaluation of Form-Based Handprint and OCR Systems*, NIST Internal Report 5932, December 1996 47
16. R. Bippus, V. Märgner: *Data Structures and Tools for Document Database Generation: An Experimental System*, Proc. Third Intern. Conf. on Document Analysis and Recognition (ICDAR), Montréal, Canada, 1995, pp. 711–714 54

This work is partly supported by the German Ministry of Education, Science and Research (BMBF) under the contract number 01IN503L/1.