Line Segmentation Metric - v3

Proposal by URO

Tobias Grüning

July 8, 2016

1 Input

Let $\mathcal{P}$ define the set of all polylines.
As input we get a set of pairs of sets $\{(\mathcal{G},\mathcal{R})_1,...,(\mathcal{G},\mathcal{R})_N\}$ where N is the number of pages in the benchmark set. Hence, there is a pair $(\mathcal{G},\mathcal{R})$ for each page, with $\mathcal{G} \subset \mathcal{P}$ the set of reference polylines (describing the baselines, which are not well-defined polylines upon which most letters "sit" and below which descenders extend [starting from the left-bottom of the oriented textline independent of the reading direction])

$$\mathcal{G} = \{\mathbf{g}_1,...,\mathbf{g}_M\}$$

and $\mathcal{R} \subset \mathcal{P}$

$$\mathcal{R} = \{\mathbf{r}_1,...,\mathbf{r}_K\}$$

the set of polylines found by the algorithm to evaluate ($\mathbf{g}_i \in \mathcal{P}\ \forall i;\ \mathbf{r}_j \in \mathcal{P}\ \forall j$).

1.1 Data Post-Processing

As such a polyline can contain an arbitrary number of points, we "normalize" all polylines $norm(\mathbf{p},k) = \mathbf{p}^*$ to standardize them. Each two adjacent points in the normalized polylines have a distance of k in x- or y-direction (dependent of the text orientation).

Example 1 (Norm-Example)**.** Let $\mathbf{p} = \{(0,0),(6,0)\}$ be a polyline. The normalized version for $k = 2$ version looks like $\mathbf{p}^* = \{(0,0),(2,0),(4,0),(6,0)\}$.

Remark 2 (Computational Issue)**.** The Parameter $k > 1$ is to reduce the computational complexity.

As a baseline is not entirely unique, and because its not crucial whether an algorithm finds it exactly or a few pixels away, some kind of tolerance value t_i has to be taken into account, for each polyline in the groundtruth $\mathbf{g}_i$. The tolerance value t_i is some kind of a border. Points of the truth and reco polyline with a distance less than t_i are counted as hits. For distances between t_i and $3t_i$ the hit value linearly decreases from 1 to 0. For distances greater than $3t_i$ it is seen as a miss.
This leads to data:

$$\mathcal{G}^* = \{\mathbf{g}^*{}_1,...,\mathbf{g}^*{}_M\}$$
$$\mathcal{R}^* = \{\mathbf{r}^*{}_1,...,\mathbf{r}^*{}_K\}$$
$$\mathcal{T} = \{t_1,...,t_M\}$$

which is processed in the next steps.

Remark 3 (Tolerance Value)**.** Since it is not entirely clear, what tolerance value has to be chosen, the whole evaluation is performed for various tolerance values within a meaningful

range (in the tool it is set to a default range of $[10, 30]$). This leads to metric values for various tolerance values. The results are displayed graphically, and the final result is the average of all these values.

2 Evaluation

To evaluate the costs we use an adaptation of the classical *precision, recall, F-measure* methodology following [1].
To define precision and recall we need some kind of "counting" function $cnt : \mathcal{P} \times \mathcal{P} \times \mathbb{R} \to \mathbb{R}$ see Algorithm 1.

Algorithm 1 count hits-algorithm

```
1:  procedure CNT(p, q; t)
2:      c ← 0
3:      for p = (p_x, p_y) point of p do
4:          d_min ← min_{q∈q}(‖p − q‖_2)
5:          if d_min ≤ t then
6:              c ← c + 1
7:          else if d_min ≤ 3t then
8:              c ← c + (3t − d_min)/(2t)
9:          end if
10:     end for
11:     return c
12: end procedure
```

This algorithm counts the number of points of $\mathbf{p}$ for which there is a point of $\mathbf{q}$ with a distance less than t (furthermore a smooth transition is performed for a distance less than $3t$).

In contrast to the first proposal (taking into account the concerns of NCSR, who mentioned that over- and undersegmentation errors are penalized to hard) I propose a slightly different interpretation of the *recall* value.
In the first proposal the *recall* value was calculated on line level. Therefore an alignment was necessary. An undersegmentation as well as an oversegmentation led to bad *recall* values (but actually the whole text is found and searchable in this scenarios). Therefore I propose the calculation of the *recall* value not on line level. Actually the counting process is evaluated with one groundtruth line $\mathbf{g}_i^* \in \mathcal{G}^*$ and the WHOLE set of found lines $\mathcal{R}^*$. Therefore the *recall* value only shows how much of the groundtruth line is found, no matter whether its under- or oversegmented. The *recall* values for all truth lines are

calculated as follows

$$rec(\mathbf{g}_i^*, \mathcal{R}^*) = \frac{cnt(\mathbf{g}_i^*, \mathcal{R}^*; t_i)}{|\mathbf{g}_i|} \quad i = 1, ..., |\mathcal{G}^*| \tag{1}$$

Remark 4. $cnt(\mathbf{g}_i^*, \mathcal{R}^*; t_i)$ only differs from Alg. 1 in Line 4. Here the minimum is not calculated over a single found polygon $\mathbf{q}$ but the whole set $\mathcal{R}^*$.

Remark 5. Following this scheme the *recall* value is an indicator showing how promising a search would be on the extracted lines.

In my point of view the *precision* value should stay as it is. The *precision* value should be an indicator how accurate the extracted baselines are, taking into account also layout issues like over- and undersegmentation. Therefore, it is essential to find the layout exactly.
I would suggest the following scheme (which is only slightly different from the one of the first proposal):

With *cnt* we can define the *precision* function (on textline level) for a groundtruth polyline $\mathbf{g}_i^*$ (and the corresponding tolerance value t_i) and a result polyline $\mathbf{r}_j^*$ as follows

$$prec(\mathbf{g}_i^*, \mathbf{r}_j^*) = \frac{cnt(\mathbf{r}_j^*, \mathbf{g}_i^*; t_i)}{\left|\mathbf{r}_j^*\right|}.$$

Given a Pair $(\mathcal{G}^*, \mathcal{R}^*)$ we can calculate a (*precision*)costmatrix $\mathbf{C} \in \mathbb{R}^{M \times K}$ with $c_{ij} = prec(\mathbf{g}_i^*, \mathbf{r}_j^*)$. Based on this matrix "best matching" pairs $\mathcal{M}$ are calculated, see Algorithm 2.

Algorithm 2 matching procedure

procedure MATCH($\mathbf{C}, \mathcal{G}^*, \mathcal{R}^*$)
  $\mathcal{M} \leftarrow \emptyset$
  $\mathbf{C}' \leftarrow \mathbf{C}$
  while $\mathbf{C}'$ is not empty **do**
    $m \leftarrow$ one of the maximal elements of ($\mathbf{C}'$)
    if $m > 0$ **then**
      //create a new matching pair
      $\mathbf{g}^* \leftarrow$ element of $\mathcal{G}^*$ belonging to m
      $\mathbf{r}^* \leftarrow$ element of $\mathcal{R}^*$ belonging to m
      $\mathcal{M} \leftarrow \mathcal{M} \cup (\mathbf{g}^*, \mathbf{r}^*)$
      $\mathbf{C}' \leftarrow$ take $\mathbf{C}'$ and delete row belonging to $\mathbf{g}^*$ and column belonging to $\mathbf{r}^*$
    else
      return $\mathcal{M}$
    end if
  end while
  return $\mathcal{M}$
end procedure

Taking into account (1) and the matching pairs $\mathcal{M}$ the *precision* and *recall* values for a given pair $(\mathcal{G}^*, \mathcal{R}^*)$ (a single page) are calculated following:

$$prec_{page}(\mathcal{M}, \mathcal{R}^*) = \frac{\sum\limits_{(\mathbf{g}^*, \mathbf{r}^*) \in \mathcal{M}} prec(\mathbf{g}^*, \mathbf{r}^*)}{|\mathcal{R}^*|} \tag{2}$$

$$rec_{page}(\mathcal{G}^*, \mathcal{R}^*) = \frac{\sum\limits_{\mathbf{g}^* \in \mathcal{G}^*} rec(\mathbf{g}^*, \mathcal{R}^*)}{|\mathcal{G}^*|} \tag{3}$$

The global *precision*, *recall* values are calculated by just averaging over the values of the different pages.

$$prec_{glob} = \frac{\sum_{i=1}^{N} prec_{page_i}}{N} \tag{4}$$

$$rec_{glob} = \frac{\sum_{i=1}^{N} rec_{page_i}}{N} \tag{5}$$

Remark 6. By calculating the average ALL pages are of same importance, no matter whether there are many or only a few lines written on it.

Remark 7 (Thresholded Version)**.** CVL suggested to implement a thresholded version of *precision* and *recall* to provide a possibility to analyse the behaviour of the investigated

algorithm. Therefore (2), (3) (thus (4), (5)) are also evaluated with

$$prec_{thr}(\mathbf{g}_i^*, \mathbf{r}_j^*) = \begin{cases} 1 & prec(\mathbf{g}_i^*, \mathbf{r}_j^*) > thr \\ 0 & else \end{cases}$$

$$rec_{thr}(\mathbf{g}^*, \mathcal{R}^*) = \begin{cases} 1 & rec(\mathbf{g}^*, \mathcal{R}^*) > thr \\ 0 & else \end{cases}$$

for different values $thr \in [0, 1]$.

$F - meas$ is calculated (in all scenarios) in the common way

$$F - meas_{page} = \frac{2 rec_{page} prec_{page}}{rec_{page} + prec_{page}}$$

3 Examples - Using the Tool

Remark 8 (PageXML)**.** The tool now accepts pageXML as input format.

In the following the behavior of this metric is shown for an example page with different recognition results. Shown in blue are the truth polylines in red the reco polylines. The Tolerance Chart shows the metric behavior for different *tol* values.

3.1 Single Pages

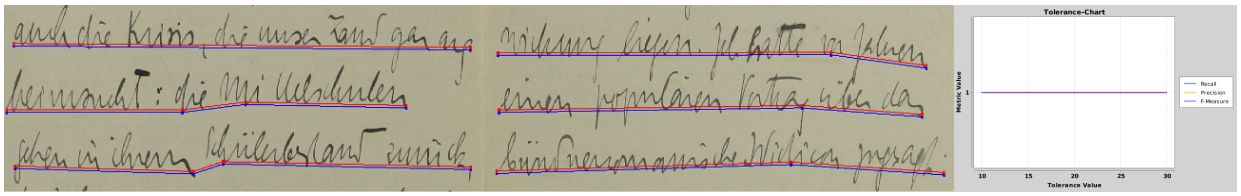

Figure 1: $prec = 1.0,\ rec = 1.0,\ f - meas = 1.0$

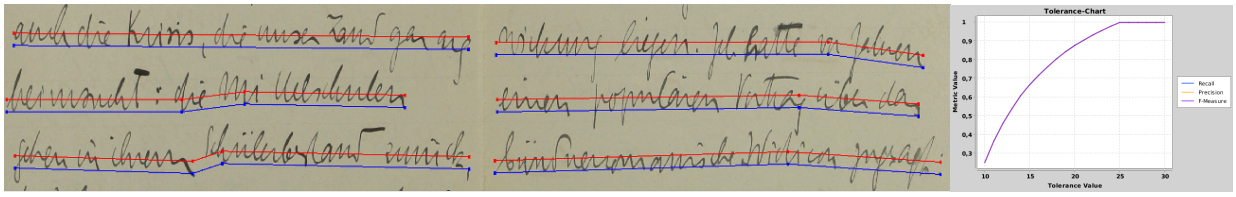

Figure 2: $prec = 0.7935,\ rec = 0.7935,\ f - meas = 0.7935$

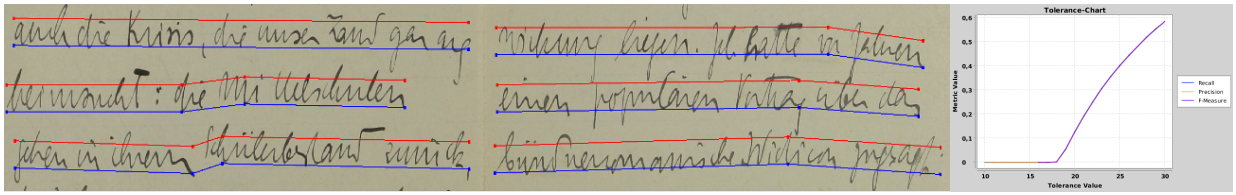

Figure 3: $prec = 0.2027,\ rec = 0.2038,\ f - meas = 0.2032$

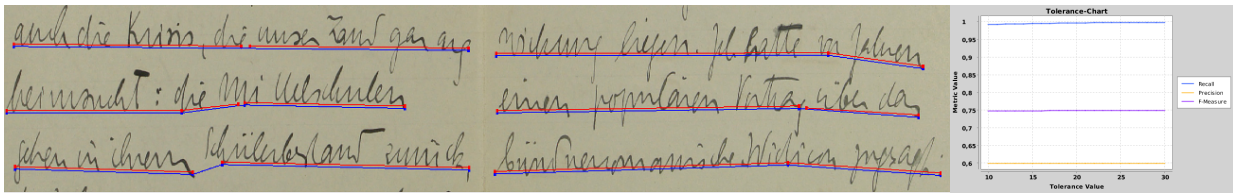

Figure 4: Oversegmentation $prec = 0.6,\ rec = 0.9968,\ f - meas = 0.7491$

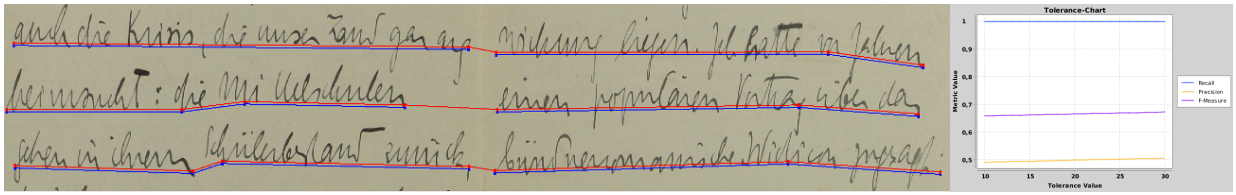

Figure 5: Undersegmentation $prec = 0.4986,\ rec = 1.0,\ f - meas = 0.6654$

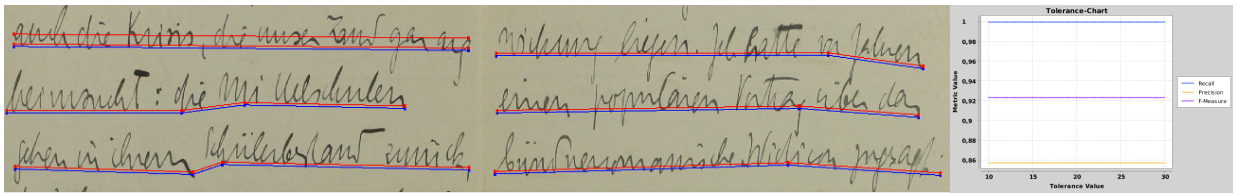

Figure 6: $prec = 0.8571,\ rec = 1.0,\ f - meas = 0.9231$

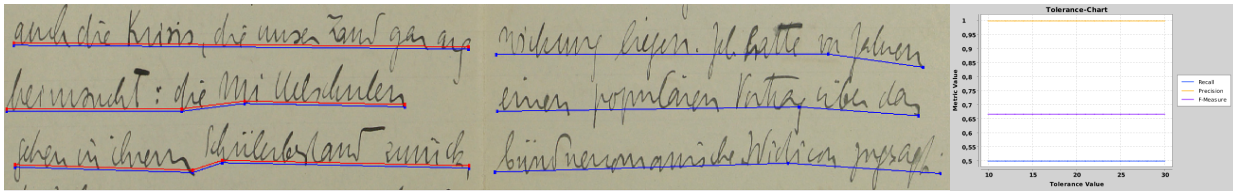

Figure 7: $prec = 1.0,\ rec = 0.500,\ f - meas = 0.6667$

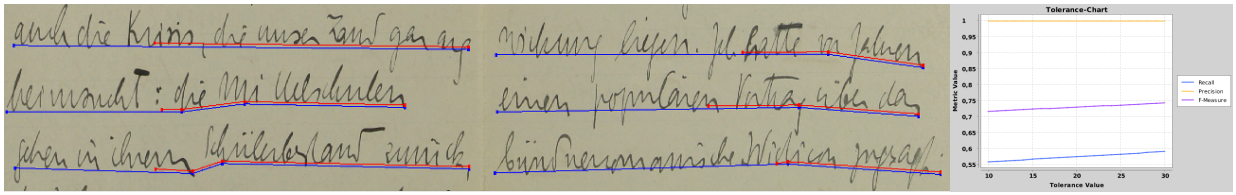

Figure 8: $prec = 1.0,\ rec = 0.5746,\ f - meas = 0.7298$

3.2 Whole Set

Averaging the results shown in Fig. 1-8 produces values

- $prec = 0.744$
- $rec = 0.7586$
- $f - meas = 0.7512$

and thresholded/tolerance charts:

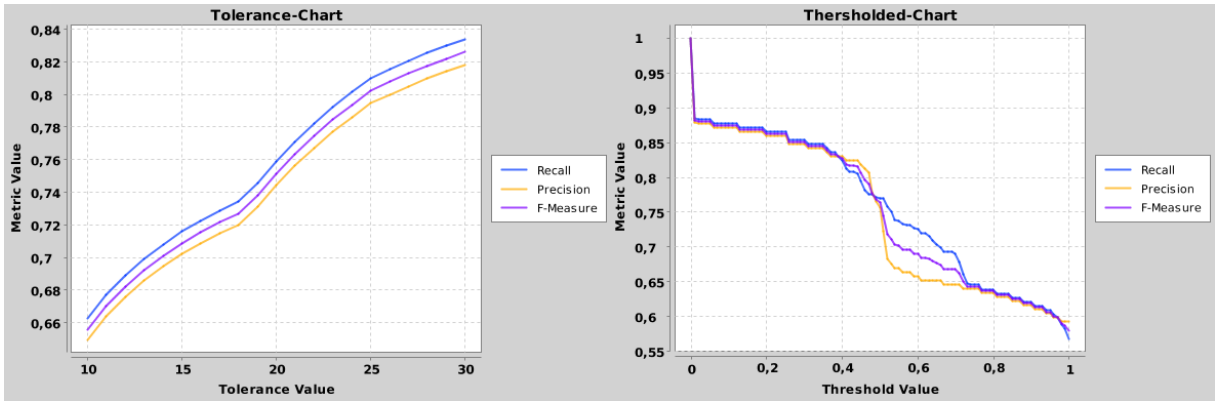

Figure 9: Tolerance/Thresholded (moving *thr*) Charts for the whole set (8 pages)

4 Metric Eval Tool

I have implemented a first prototype of a tool. It is written in java. Assuming that java is installed on your system it is a standalone application. As shown in Fig. 10 it is simple to use.

5 Discussion

There were some additional points mentioned by UPVLC in Innsbruck.
Concerning the issue of a non-perfect alignment I would like to mention, that the matrix **C** in meaningful scenarios tends to be sparse! Taking a reco line, only for very few truth lines the *precision* value is greater than 0. Therefore in a later version the calculation of a perfect matching without the assumption of a natural order should be possible in

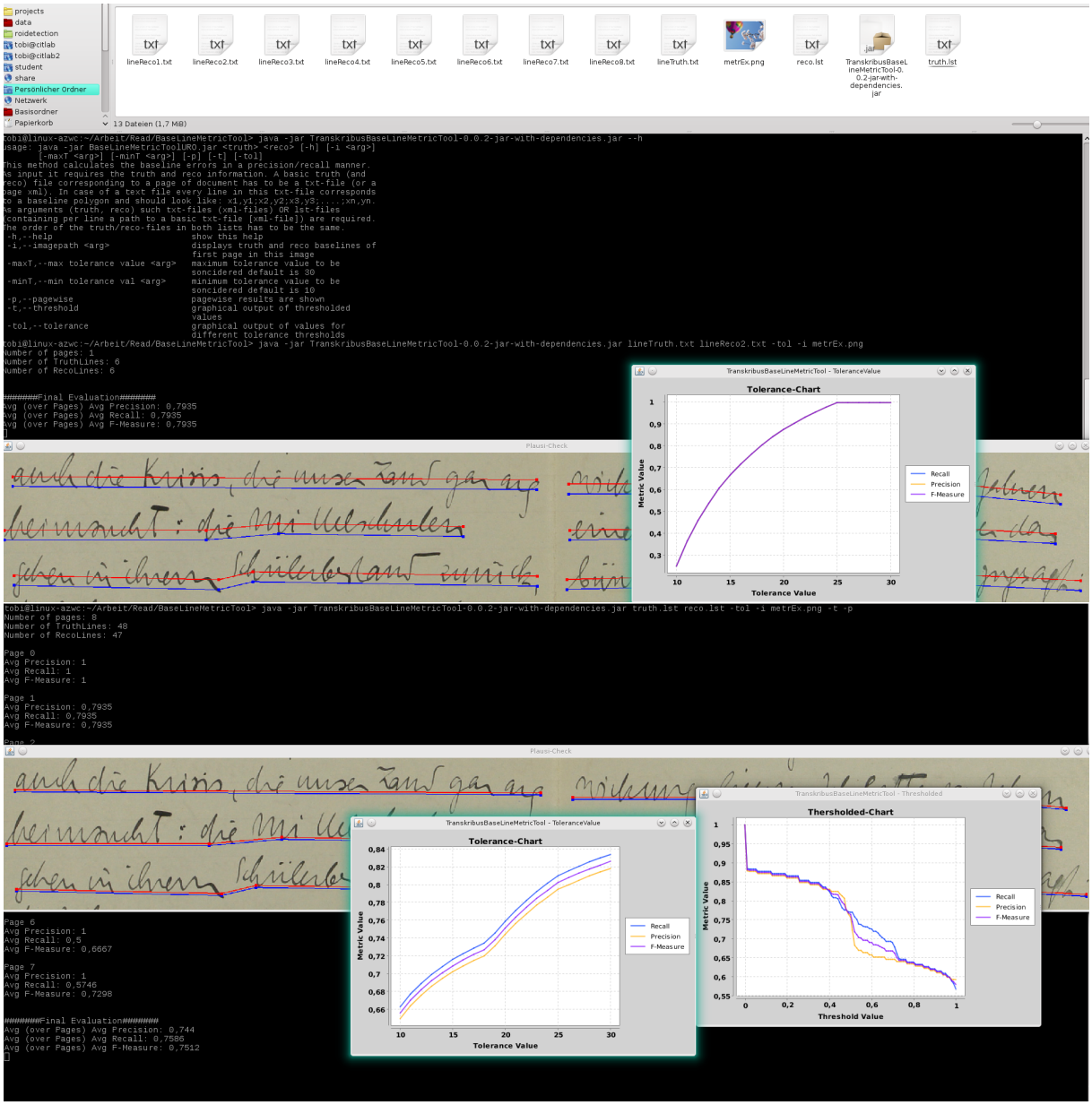

Figure 10: BaseLineMetricToolURO

meaningful time complexity. Nevertheless, in my opinion this will not have a big impact at all.

References

[1] B. Gatos, N. Stamatopoulos, and G. Louloudis. Icdar 2009 handwriting segmentation contest. In *Document Analysis and Recognition, 2009. ICDAR '09. 10th International Conference on*, pages 1393–1397, July 2009.