UnShredder – Interim Report

1. INTRODUCTION

Ever since the paper shredder was invented, people have worked on sticking the pieces back together again. Recently, techniques permitting the purely electronic storage and transmittal of sensitive documents have been developed but, because of convenience or for legal reasons, many sensitive documents are still printed and eventually shredded.

The recent surge of interest in the area of reconstructing shredded documents has shown that many people overestimate the security offered by the shredder. One example is the 2011 DARPA shredding challenge, which offered a $50,000 prize for the first team to successfully solve a series of puzzles printed on 5 shredded documents. When a winner was declared after 32 days, the performance was generally met with amazement from the general public, especially since the final puzzle in the contest (see Figure 1) is on par with documents shredded by some of the most expensive devices available today.

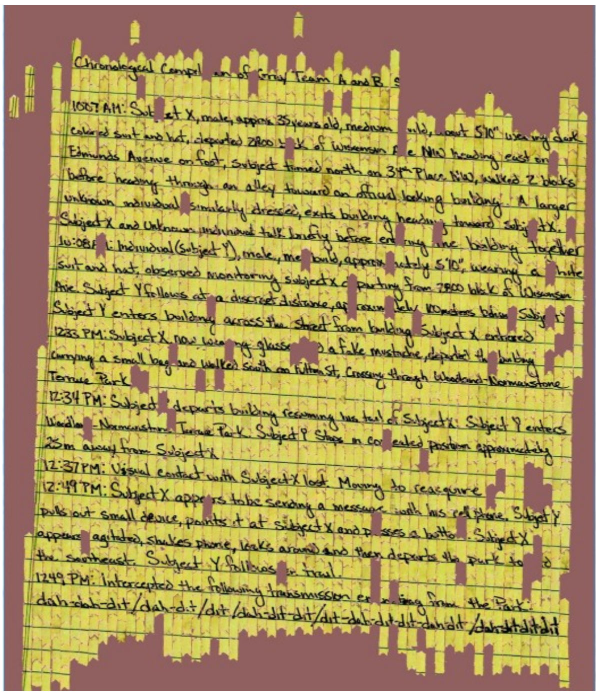


Figure 1. DARPA Challenge, reconstruction of part of the hardest puzzle

The DARPA Challenge itself was motivated by the difficulty troops encounter in war zones while trying to make sense of the remnants of destroyed documents, and further research in this area could have a significant impact on a number of related problems in the fields of forensics and investigative science. One of the most notable projects that could benefit from advancements in unshredding technology is the current effort to recover the shredded archives of the East German secret police. These consist of a total of 16,000 bags of shredded documents and, so far, it took three dozen people six years to reconstruct 300 of them. At that rate, it would take 11,520 person-years to finish the task.

This paper looks at the challenges involved in the automatic reconstruction of both strip (vertically cut) and cross-cut (vertically and horizontally cut) shredded documents. A novel probabilistic score function is proposed and shown to perform well in comparison with the standard cost functions used in literature. Additionally, several tractable search heuristics are analysed. The paper attempts to identify which aspect of the reconstruction pipeline is currently the bottleneck and therefore which element can most benefit from future work. The security currently offered by several types of shredders is also touched upon.

2. LITERATURE REVIEW

In previous work [1, 2, 3, 4, 5], the solution to the reconstruction problem has been formulated with the help of two functions.

A cost function is used to represent how well two pieces match, the values varying from 0 for a perfect match to infinity for an impossible one. A search function is then used to find a permutation that minimizes the sum of the costs of all adjacent pieces. This permutation is considered the most likely solution.

Significant effort has been made towards finding a good search function. In [1] it is shown that the problem is NP hard by reduction to the Travelling Salesman Problem. In [2] an exact solution is attempted via Integer Linear Programming which yields very good results but is intractable for any number of strips above 150. Furthermore, several attempts [3, 4] have been made at applying evolutionary algorithms to the search problem.

In contrast, relatively little progress has been made in developing the cost function. Most papers settled on a simple formulation which does a weighted difference of each pair of adjacent pixels on either side of the proposed join and increases the cost of the join if the pixels are too dissimilar. See [1] for details.

One refinement for the cost function is proposed in [5], where the authors notice that the previous formulation places too much emphasis on matching adjacent white pixels which can obscure the real solution (see Figure 2). As such they base the cost on how well the black pixels match and also add a heuristic which looks at the positioning of the rows of text in the proposed match.

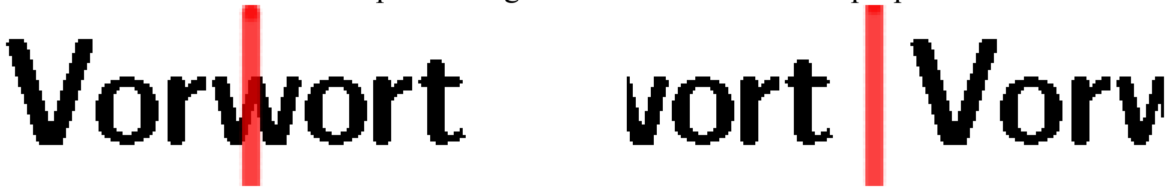


Figure 2. Left: the correct match. Right: white-on-white match which would get a perfect cost under the cost function proposed in [1] and be incorrectly preffered

The authors of [6] present another possible cost function improvement by looking at way computer vision methods might be applied to identify individual characters that were split between different shreds. The paper has promising preliminary results but doesn't provide comparisons with other cost functions, so aspects such as robustness to noise remain unclear.

An additional problem encountered in document reconstruction is that often the shreds of many different documents will be mixed together thus making the reconstruction process completely intractable without first applying a clustering step that separated the pieces into likely document classes. Towards this goal [7] analyses the use of MPEG-7 standard descriptors towards the classification task, obtaining encouraging results especially for the colour descriptors. A different approach is taken in [8] where the authors propose a number of custom features for the clustering, such as paper type, writing style, colour extraction and line detection.

3. PROBABILISTIC SCORE

This paper proposes a departure from the previous cost function definitions, looking instead at using a probabilistic model to directly estimate the likelihood of two edges matching.

The central idea is to estimate the probability that a candidate pixel is correct given several of its neighbour pixels, called the pixel's context (see Figure 3). These conditional probabilities are estimated by analysing the distribution of pixels in the shreds the algorithm is trying to piece together, which should be representative of the distribution in the reconstructed document.

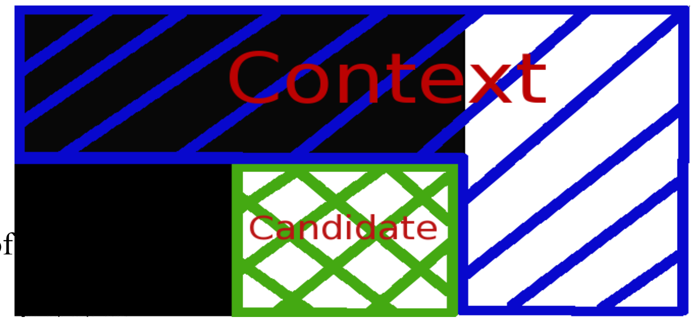


Figure 3. The probabilistic model estimates the probability that the candidate pixel is white or black based on the four context pixels

Once these conditional probabilities are known, the probability of two edges matching can be calculated by sliding the context down the proposed edge and multiplying all the individual candidate pixel probabilities. Additionally, there is exactly one correct match for every edge, so the sum of the probabilities of all matches along one edge should sum up to one. In order to satisfy this condition the probabilities are normalized along every edge, which ends up mitigating the predisposition towards whitespace that some of the other cost functions suffer from. This is because there are usually many white edges available at any time and the probability mass will therefore be diluted between them. (see Table 1). To further discount the white on white matches we can introduce dummy white pieces into the shreds pool.

	Raw probabilities			Normalized probabilities		
	Piece 1	**Piece 2**	**Piece 3**	**Piece 1**	**Piece 2**	**Piece 3**
Edge 1	1.00	1.00	1.00	0.33	0.33	0.33
Edge 2	0.50	0.10	0.10	0.71	0.14	0.14

Table 1. After normalization, the match between edge 2 and piece 1 is considered a better bet than any match containing edge 1, as edge 1 could match equally well with pieces 1, 2 or 3

Preliminary results suggest that this probabilistic score compares favourably to the cost functions used in [1] and [5] (see Figure 4).

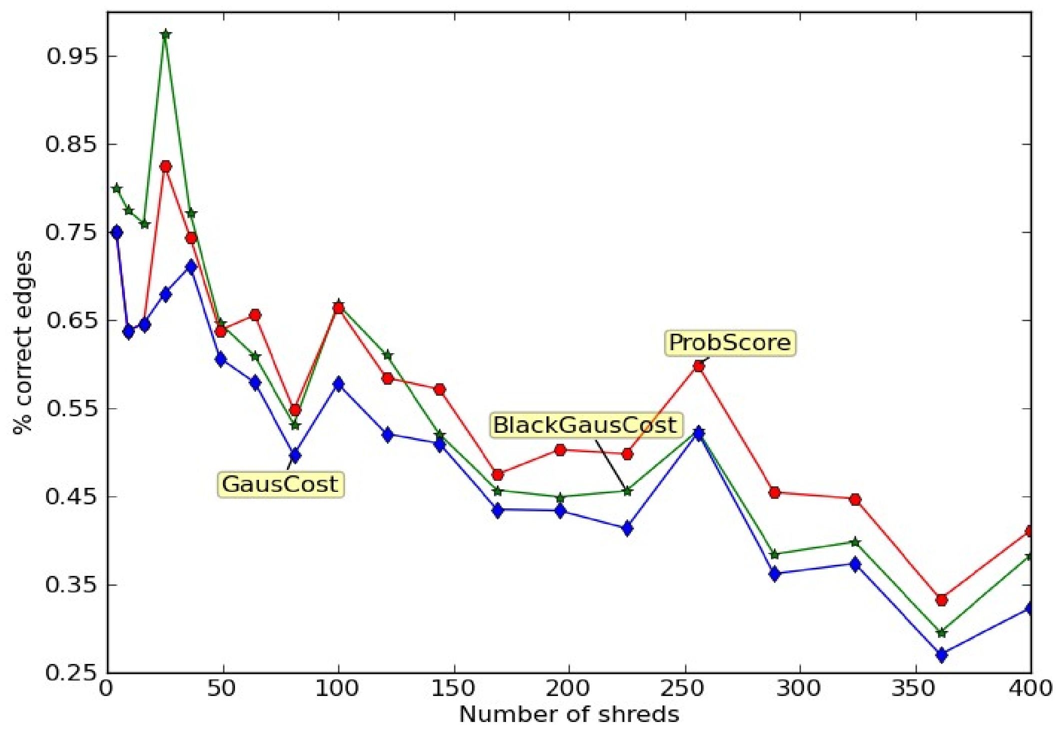
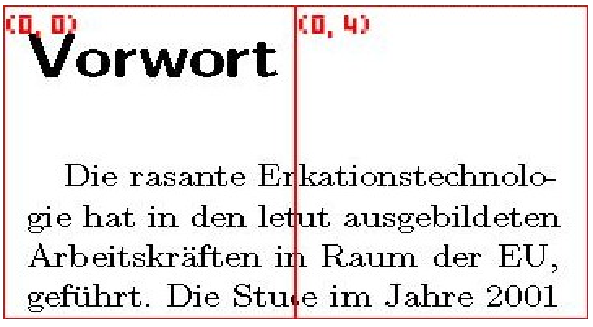


Figure 4. The probabilistic score has better results on medium and large instances than the cost functions presented in [1] (GausCost) and [5] (BlackGausCost)

An additional benefit of this probabilistic score function formulation is that it is modular and allows for the easy incorporation of other probabilistic models looking at different kinds of features. This is an important aspect as on more secure shredders more than one type of feature will likely be necessary for the creation of a satisfactory cost/score function. As the size of the shreds decreases it becomes increasingly difficult to define a good cost/score function which considers only the edge pixels. (see Figure 5)


(0, 0)
(0, 4)
Vorwort

Die rasante Erkationstechnolo-
gie hat in den letut ausgebildeten
Arbeitskräften in Raum der EU,
geführt. Die Stu‹e im Jahre 2001

Figure 5. An incorrect match that would be very difficult to detect by a cost/score function which only looks at the edge pixels

4. HEURISTIC SEARCH

Extending the graph-based heuristics introduced in [5], this paper looks at a Kruskal variant which tries to always pick the most probable edge and match the two pieces along that edge. This method creates multiple clusters of pieces which it must be able to later merge (see Figure 5). If the best edge it has found would lead to a merge of two clusters, the potential match is checked for a superimposition of pieces, which would lead to the merge being rejected.

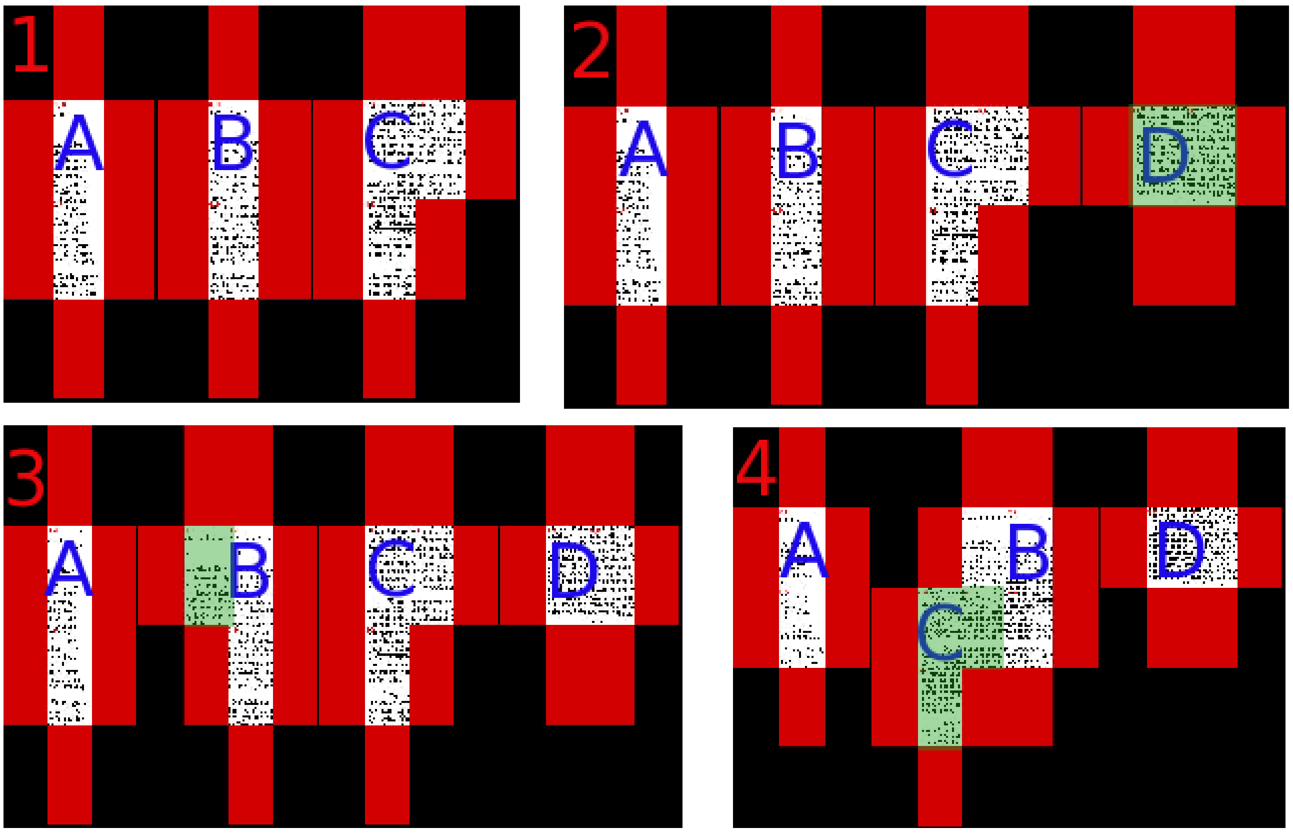


Figure 5. Four steps from the middle of a Kruskal search are shown. The clusters are called A, B, C and D, the green pieces highlight the changes that occurred in every step and the red pieces show the positions that the search considered for the insertion of a new piece.

Step 1: there are three clusters. Step 2: a fourth cluster is created by adjoining two shreds. Step 3: a shred is added to the pre-existing cluster B. Step 4: the clusters B and C are merged, thus reducing the number of clusters to three again.

A problem with the Kruskal heuristic presented here as well as the heuristics from [5] is that they cannot correct previous errors and therefore are prone to a cascading effect (see Figure 5). Essentially if an error is made early in the search process this can have a major effect on the result as the search is now trying to find shreds which match well with the wrongly placed piece. This problem only gets worse as the number of shreds increases (see Figure 5).

In order to address this problem, several error correcting mechanisms were analysed. The simplest approach is to consider pieces that have already been placed as movable and treat them the same as the pieces which haven't been placed. This basic approach can however lead to an infinite cycle if it happens that one piece is the best match for two different edges, as the greedy algorithm will continue to switch it between the two. This specific problem can be solved by giving the greedy correction algorithm a lookahead of size one, by only moving a piece if this would increase it's probabilistic score. This solution however will only eliminate cycles of length 2, in order to eliminate all cycles in this way would require a complete examination of the search tree from that point onwards, which quickly becomes intractable.

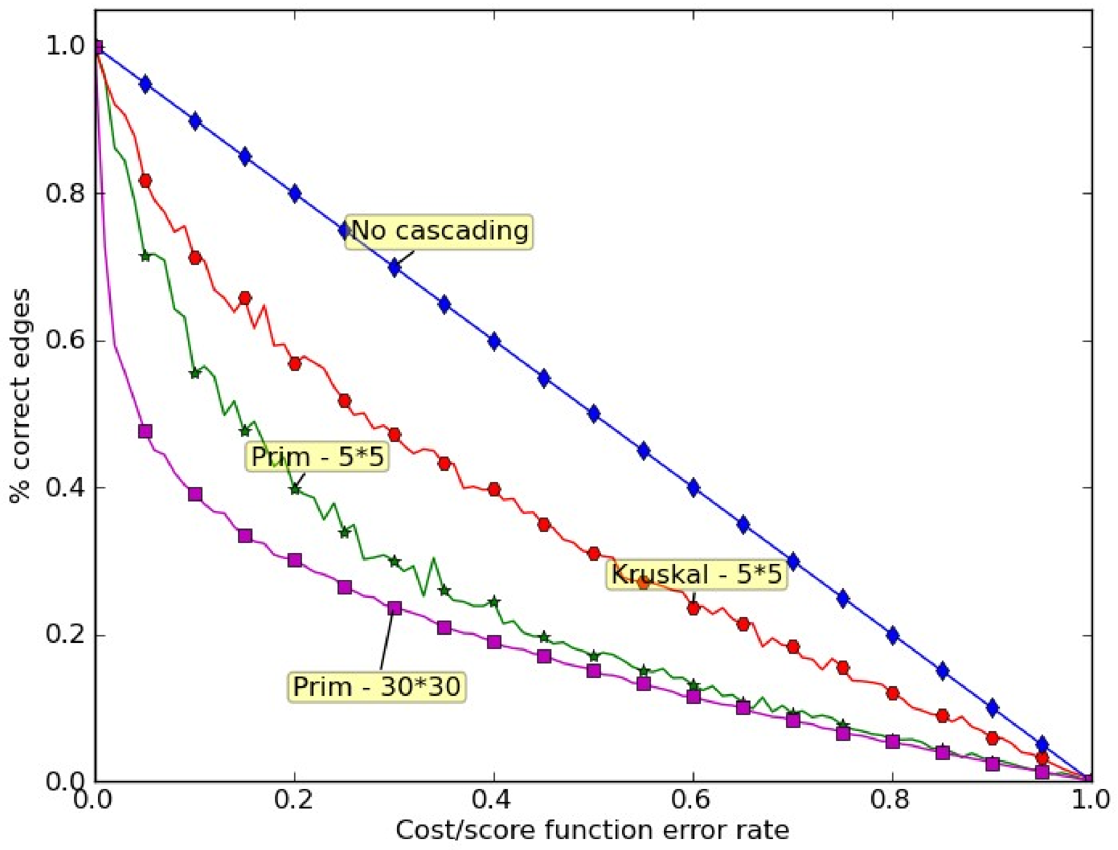


Figure 5. The effect the error in cost/score has on the final error of the method for both search heuristics on 5*5 shreds and for Prim on 30*30 shreds

In order to eliminate cycles while using a fixed size lookahead, all previous states that the search has passed through can be remembered and therefore cycles can be detected. Once a cycle is detected the algorithm can choose to revert to the best state encountered within the cycle and then pick a move that breaks the cycle.

Another problem that plagues the heuristic search methods presented here is that while the search is in progress it cannot be known which pieces will end up on an edge and which will be in the centre. This means we cannot count the cost/score of an edge piece being adjacent to white space which in turn places no incentive on the search to find compact shapes that have less pieces on an edge (see Figure 6).

This second problem can be ameliorated by doing a post-processing step to the search. When all the pieces have been placed we finally know what the proposed edges are, so we can now keep count of the external whitespace score and apply the same search and cycle detecting process described above. However, in order for this greedy search to find correct solutions it will require a large lookahead (see Figure 6) which again becomes quickly intractable. However, since all previous states are recorded, this post-processing step is guaranteed to obtain a final state that's either better or at least equivalent to the one it started with. This guarantee cannot be made for the previous correction heuristic.

The above process also slows down the computation significantly. This is because, even though infinite cycles are eliminated, the algorithm can still make a large number of moves before it returns to a previous state where a cycle can be detected and stopped. This behaviour additionally limits the size of the lookahead that can be given and the size of the lookahead limits the performance boost.

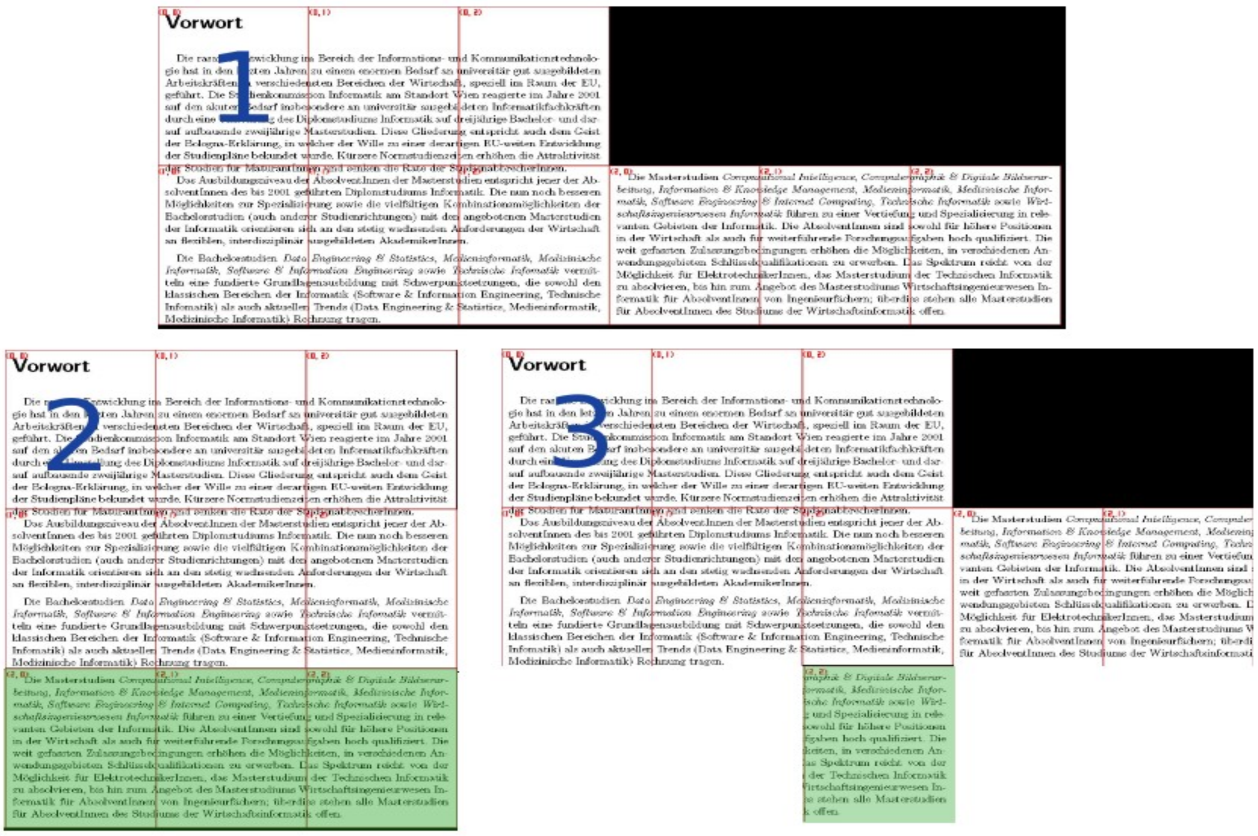


Figure 6. Since external whitespace is not counted towards the score 1 and 2 have the same score even though 1 has 16 external edges and 2 has only 12. In order to move from 1 to 2 the greedy search would have to pass through 3 which also has 16 external edges. In this case the transition to the correct result will only be made if the search can see at least 3 moves ahead

With a lookahead of 1, the corrections done during the running of the search seem prone to noise and may actually hurt the result. The post-processing step however provides a small but consistent boost in performance (see Figure 7)

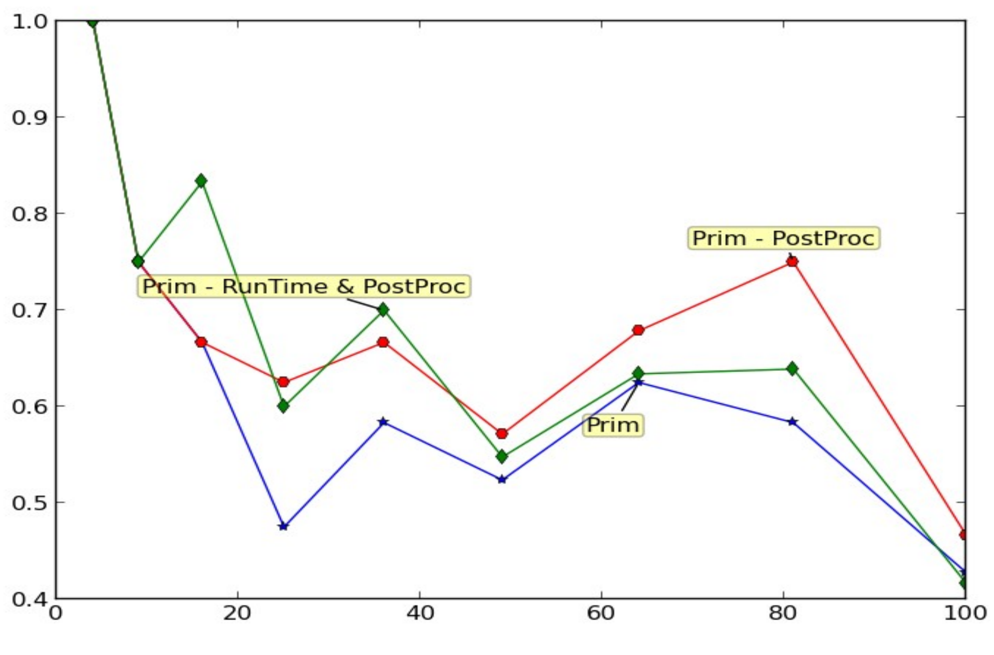


Figure 7. Comparison between the basic Prim algorithm and the enhanced versions either with just the post-processing step or with both the run-time corrections and the post-processing. The run-time corrections are not very consistent with a small lookahead

5. CONCLUSIONS

Preliminary results, which assume the correct orientation of the shreds is known a priori, are shown (see Figure 8)

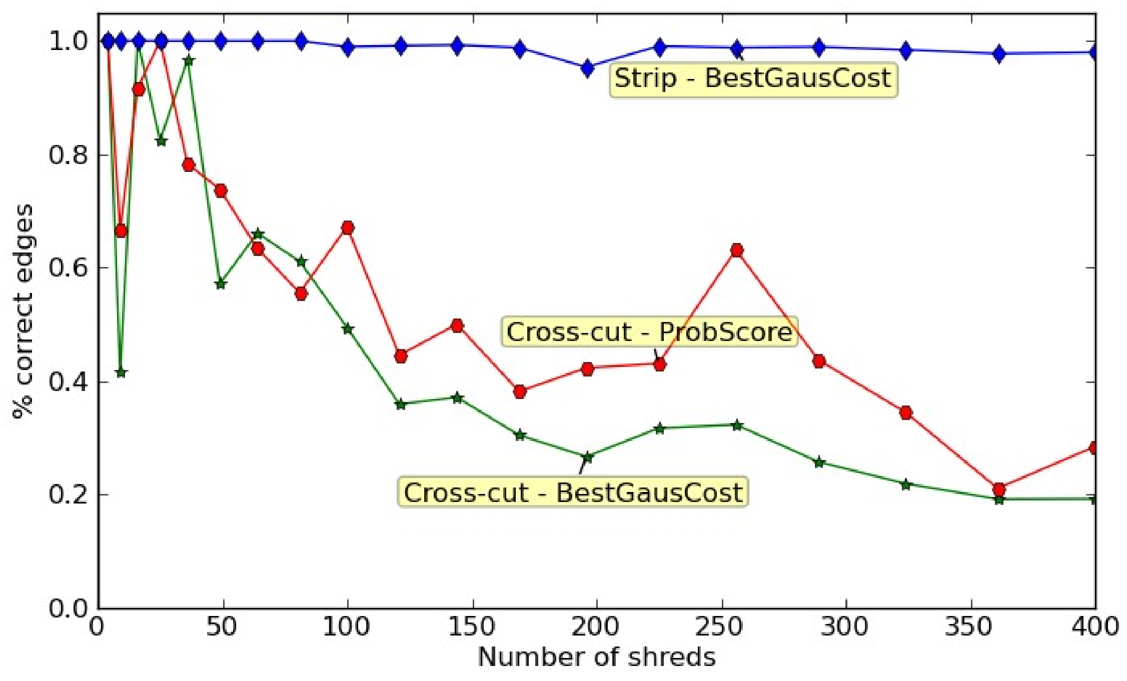


Figure 8. Performance on cross-cut shreds shown with the probabilistic score and the best performing cost function. Performance on strip shreds shown with the best cost function (the probabilistic score performs similarly, and is not shown). The search function used is Prim

The strip shred variant seems significantly easier to solve than the cross-cut version. This discrepancy can be explained by the fact that the shorter edges produced by cross-cutting are much more prone to noise because of the reduced number of edge pixels. Additionally, for text documents, horizontal cuts have a significant chance of falling between two lines of text, in which case the cost/score function has no information on how to order the lines.

6. Progress Report:

6.1. Reading:

- Initial literature review looking at the current state of the research area - DONE
- Review of published local search methods - DONE
- Identifying cost functions used in literature so far – DONE
- Analysis of benefits and drawbacks of cost functions incorporating computer vision or optical character recognition methods - DONE
- Review of performance of exhaustive search methods including integer linear programming and reduction to a Travelling salesman problem – DONE
- Analysing the possibility of using clustering techniques to reduce the problem size – Week 2

6.2. Writing:

- Summary of papers read - DONE
- First project presentation - project description and goals - DONE
- Second project presentation – analysis of a novel cost function and of several search heuristics – DONE
- "Recovering shredded documents" - abstract written for the SAC 2013 student research competition - DONE
- Interim Report – Week 2
- Poster and presentation for the SAC 2013 competition – Week 8
- Final Report – from Week 5 until deadline

6.3. Coding:

- Bit to Bit and Gaussian Cost functions - DONE
- Previous cost function restricted to using black bits - DONE
- Greedy1D search baseline - DONE
- Prim search baseline - DONE
- Gaussian simulation of edges cost - DONE
- Image processing simulating the shredding process - DONE
- Refactor - DONE
- Kruskal heuristic search function. - DONE
- Removal of image processing speed bottlenecks. - DONE
- Visualizations tools, can observe what algorithm does at every step - DONE
- Basic neural network cost function - DONE
- Cost function based on distribution of 6 pixel window taken over the shreds - DONE
- Probabilistic cost function - DONE
- Analysis of the cascading effect a mistake has with different search heuristics. - DONE
- Preliminary analysis of cost functions. Probabilistic cost seems to perform best. - DONE
- Optimization of kruskal heuristic - DONE
- Analysis of the cascading effect on new search heuristics – DONE
- Search functions which can correct previous mistakes, during search or as a post-processing step – DONE
- Probabilistic search functions – Week 2 - 3
- Developing an evaluation measure more closely correlated to how difficult it would be for a human to extract the relevant information from the reconstructed document – Week 3
- Cost function incorporating higher level features (eg: row positioning information) – Week 4
- Running cross-cut method against benchmarks used in other papers – Week 5
- Running strip-cut method on paper that is actually shredded and scanned, analysis of the effect of the extra noise – Week 6
- Comprehensive evaluation of the performance of the probabilistic cost function as compared to all cost functions present in literature – Week 7
- Comprehensive evaluation of the various search heuristics against one another and against other methods used in literature – Week 8

REFERENCES

[1] Prandtstetter, M., and Raidl, G. - Combining Forces to Reconstruct Strip Shredded Text Documents *In HM '08: proceedings of the 5th international workshop on hybrid metaheuristics (pp. 175–189). Berlin: Springer.*

[2] Prandtstetter, M. - Two Approaches for Computing Lower Bounds on the Reconstruction of Strip Shredded Text Documents *Technical Report TR–186–1–09–01, Technishe Universitat Wien, Institut für Computergraphik und Algorithmen, 2009*

[3] Prandtstetter, M., and Raidl, G. - Meta-Heuristics for Reconstructing Cross Cut Shredded Text Documents *In GECCO '09:proceedings of the 11th annual conference on genetic and evolutionary computation (pp. 349–356). New York: ACM Press.*

[4] Schauer, C., Prandtstetter, M. and Raidl, G. - A Memetic Algorithm for Reconstructing Cross-Cut Shredded Text Documents *In Hybrid Metaheuristics, pp. 103–117, 2010.*

[5] Sleit, A., Massad, Y. and Musaddaq M. - An alternative clustering approach for reconstructing cross cut shredded text documents *Telecommunication Systems, 2011 – Springer*

[6] Perl, J., Diem, M., Kleber, F. and Sablatnig, R. - Strip Shredded Document Reconstruction using Optical Character Recognition Imaging for Crime Detection and Prevention, 2011

[7] Ukovich, A., Ramponi, G., Doulaverakis, H., Kompatsiariss, Y. and Strintziss, M.G. - Shredded document reconstruction using MPEG-7 standard descriptors *Proceedings of the International Symposium on Signal Processing and Information Technology, pp. 334–337, IEEE, 2004.*

[8] *Diem, M., Kleber, F. and Sablatnig, R. - Document Analysis Applied to Fragments: Feature Set for* the Reconstruction of Torn Documents i*n Proceedings of the 9th IAPR International Workshop on Document Analysis Systems, DAS '10, pages 393–400, 2010.*