A Composable Strategy for Shredded Document Reconstruction

Razvan Ranca and Iain Murray

School of Informatics, University of Edinburgh,
10 Crichton Street, Edinburgh, Scotland, United Kingdom
ranca.razvan@gmail.com, i.murray@ed.ac.uk

Abstract. The reconstruction of shredded documents is of interest in domains such as forensics, investigative sciences and archaeology, and has therefore been approached in many different ways. This paper takes a step towards bridging the gap between previous, disparate, efforts by proposing a composable, probabilistic solution. The task is first divided into independent sub-problems, and novel approaches to several of these sub-problems are then presented. The theoretical properties and empirical performance of these novel approaches are shown to compare favourably to those of previously published methods.

Keywords: shredded document, strip-cut, cross-cut, unshred, deshred

1 Introduction

The US Federal Trade Commission recommends the shredding of sensitive documents as a good protection method against becoming one of the millions of victims of identity theft[1]. However, the successfully solved DARPA Shredder challenge[2] and the appearance of commercial document reconstruction services[3], raises questions about the current security offered by the shredder. Further research in this area will benefit projects such as the ongoing effort to recover the shredded archives of the East German secret police, which consist of 16,000 bags of shredded documents. Since it took 3 dozen people 6 years to reconstruct 300 of these bags ([5]), at this rate, the project would require 11,520 person-years.

In this paper, we approach the unshredding challenge by splitting the problem into four independent sub-problems. A *pre-processing* step transforms the noisy images of scanned shreds into uniform "ideal shreds". A *scoring function* looks at the potential match between every pair of ideal shreds and indicates the viability of that match. A *search method* is then employed to find a placement of the ideal shreds in 2D space that obtains a good global score, where the global score is the sum of the local scores determined by each pair of neighbouring shreds. Finally, a *user-interaction* step can be weaved in with any of the above as to help catch errors before they have a chance to propagate.

[1] http://consumer.ftc.gov/features/feature-0014-identity-theft
[2] http://archive.darpa.mil/shredderchallenge/
[3] eg: http://www.unshredder.com/

In Sections 3 to 5, we will present: a novel, composable and probabilistic scoring function; an original, graph-inspired, search heuristic which balances speed and accuracy while achieving modularity; and a tractable up/down orientation detection method which can be used as part of the pre-processing step.

2 Related Work

Significant effort has been made towards finding a good search function. In [9] the authors show that the search is NP hard by reduction to the Travelling Salesman Problem. Prandtstetter ([8]) attempts an Integer Linear Programming solution which yields good results but is intractable for more than 150 shreds. In [10, 11], good results are obtained by applying evolutionary algorithms to the problem.

In contrast, relatively little progress has been made in developing the score function. In [2, 8–11], the authors settled on a formulation which selects a score based on a weighted difference of the adjacent pixels on either side of the proposed join ([2] provides a formal definition of this method). Sleit, Massad and Musaddaq ([13]) refine this formulation by placing an emphasis on black pixels, thus discounting the information content of white pixels. In [7], the authors try a different approach, by employing optical character recognition techniques. Their method is however left as a proof-of-concept, since it is not integrated with a search function or evaluated against any other scoring methods.

Pre-processing can be split into several independent functions, which have been previously explored. Skeoch ([12]) shows how to extract the shreds from scanned input images via rectangle and polynomial fitting. The authors of [3] fix the skew of the extracted shreds by framing the issue as an optimisation problem and finding the best rotation angle for each shred. Up/down orientation of documents is explored in [1, 4] with good results, though the methods are only evaluated on full documents, not shreds.

Finally, user-interaction has been approached in many ways. For instance, in [3, 13], the authors allow the users to mark correct and incorrect edges between successive runs of the search method, while in [14] the scoring function is influenced by the user drawing what the neighbour of a shred might look like.

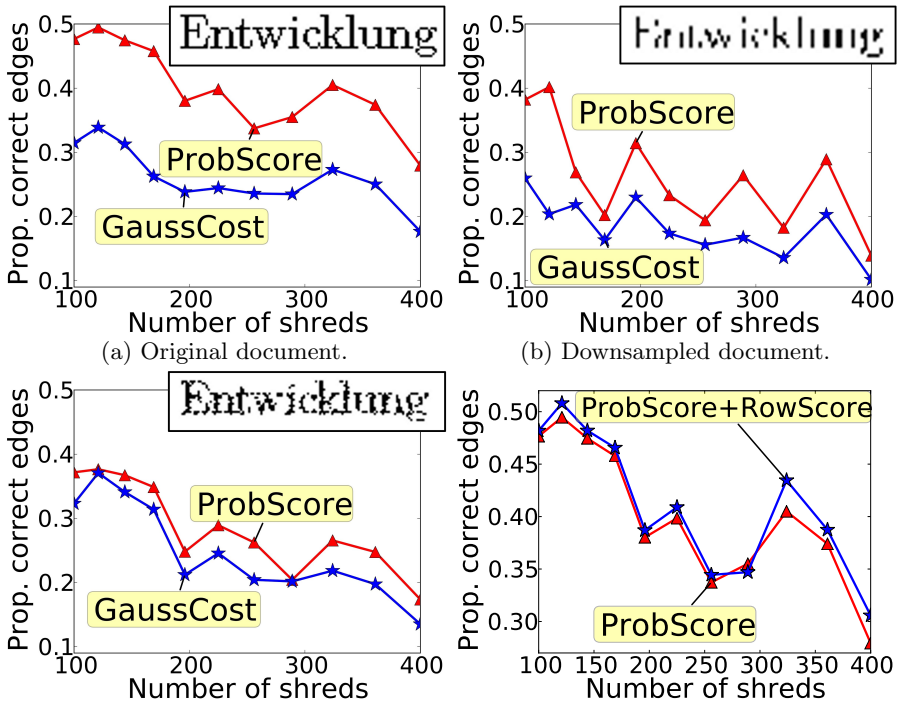
3 Probabilistic Score

We propose a novel score function formulation, which uses a probabilistic model to directly estimate the likelihood of two edges matching. Employing a probabilistic model offers several advantages, such as an increase in robustness given by the ability to train the model on the given document shreds and an ease of composability with other models, which can be achieved by simply multiplying the different predicted probabilities and re-normalising the results.

We test this idea by implementing a basic probabilistic model, which is based on the conditional probability that a pixel is white or black given a few of its neighbouring pixels. Formally, given edge *Et*, *ProbScore* returns the best match for Et and the probability of that match. ProbScore is defined as:

procedure PROBSCORE(Et)
  Initialise ps ▷ *probabilities of matches, initially all 1*
  for all $Ex \in Edges$ **do**
    for all $pixel \in Ex$ **do**
      $ps_{Ex} \leftarrow ps_{Ex} * \Pr(pixel|Neighbours_{Et}^{pixel})$
  Normalise ps ▷ *probabilities must sum up to 1*
  return $\arg\max ps$, $\max ps$

Empirical results show that this probabilistic score compares favourably to the most common scoring function used in literature ([2]), both on noise-less documents and on several noisy documents (see Figure 1). Additionally, in order to showcase the benefits offered by the composable nature of ProbScore, we combine it with another probabilistic model called *RowScore*. RowScore calculates the distance between rows of text in neighbouring shreds and applies a Gaussian model to this distance. Even such a simple probabilistic model gives ProbScore a small but consistent boost (see Figure 1). In a similar way, ProbScore could be composed with more complex models, such as that proposed in [7].



(a) Original document. (b) Downsampled document.

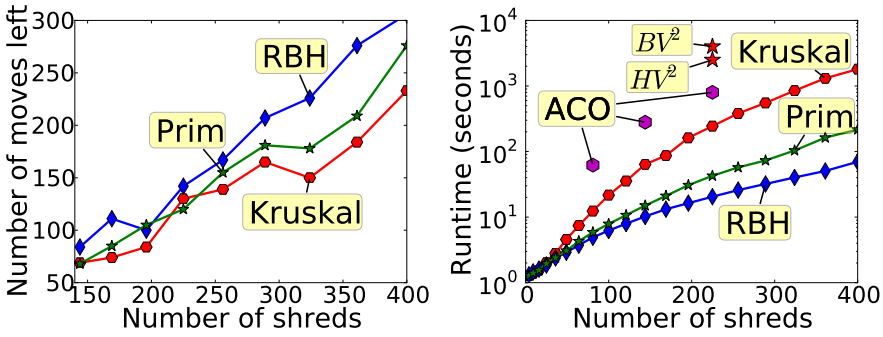
(c) Noisy document. (d) Score function composition.

Fig. 1: Figures **a**, **b** and **c** compare our method with the most common previously used function. A sample word from each document is shown in the upper right corners. Figure **d** shows the improvement obtained, on a noiseless document, by composing our function with another, very simple, probabilistic model.

4 "Kruskal-Inspired" Heuristic Search

We present a search method inspired by the minimum spanning tree "Kruskal's algorithm" ([6]). The method greedily picks the most probable edge and tries to match the two shreds along that edge. This process creates multiple clusters of shreds which will eventually be merged into a single solution. Before performing a merge, we check if the move would result in two shreds being superimposed, in which case the merge is aborted.[4]

The Kruskal heuristic outperforms previously proposed bottom-up heuristics, but is still significantly more tractable[5] than any of the top-down optimising search functions (see Figure 2).



(a) Performance comparison.

(b) Runtime comparison.

Fig. 2: Figure **a** shows the performance of our method compared to that of two heuristics introduced in [10]. Here the X-axis shows how many moves would have to be performed to get from the outputted solution to the correct solution. Figure **b** shows the scalability of our approach when compared to the aforementioned heuristics and a few top down optimising searches introduced in [11] (ACO is an Ant Colony Optimisation, while HV^2 and BV^2 are genetic algorithms).

A novel aspect of the Kruskal heuristic is that, if the next move is uncertain, the execution can be stopped. The early stopping mechanism is triggered when the score of the next best move is below some threshold. Once the search has been stopped we return a partial solution, which can significantly reduce the search

[4] A conceptually similar search heuristic is introduced in [13]. That formulation, however, uses a static list of scores and thus cannot be used with our probabilistic scoring function which requires the scores to be updated after every merge.

[5] The Kruskal, Prim and RBH heuristics are written in python, while the ACO, BV^2 and HV^2 optimisation methods are written in C. Additionally, the implementation of Kruskal is a non-optimised proof-of-concept. A non-naive re-implementation could bring its performance more in line to that of Prim. As such, the scalability results should be viewed as upper bounds on the performance of the heuristics.

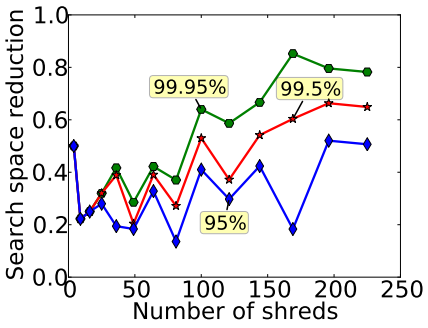


Fig. 3: The reduction in search space corresponding to 3 stopping conditions. "Search space reduction" is defined as $\frac{\text{Final no. pieces}}{\text{Initial no. pieces}}$.

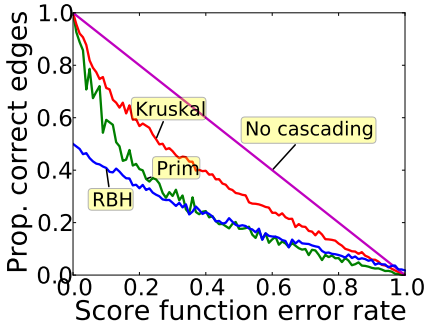


Fig. 4: The effect the error in score has on the final error of the method for the 3 search heuristics. The cascading effect causes a small error in the score to be exaggerated by the search method.

space of the problem. As seen in Figure 3, even the extremely conservative 99.95% stopping condition helps us reduce the search space to between 40% and 80% of its original size.[6] Since the complexity of the search function is exponential in the number of pieces, these reductions are significant and may allow previously intractable search methods to now be run on the smaller instance.

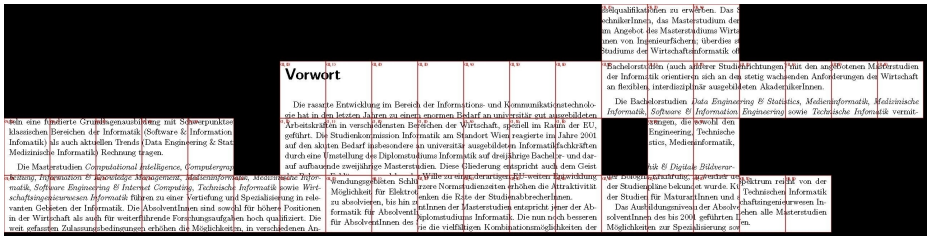
Another benefit of our Kruskal heuristic can be seen when analysing the *cascading effect*, which is an issue common to all greedy, bottom-up, heuristics. The cascading effect is caused by the inability of the heuristics to correct a wrongly placed shred. This inability means that the search will instead try to find additional pieces which match well with said, wrongly placed, shred. These pieces will therefore have a significant probability of also being wrong, thus starting a chain reaction. While cascading is a problem for all heuristics, our method proves to be more resilient than previous formulations (see Figure 4).

Finally, in Figure 5, we show some full and partial reconstructions of documents. One thing worth noticing from this example is that cross-cut documents are significantly harder to solve than strip-cut ones, even if the total number of shreds is the same. This difficulty is due to the short edges produced by cross-cutting, which are harder to model accurately. Horizontal cuts also have a significant chance of falling between two lines of text, in which case the score function has no information on how to order the lines.

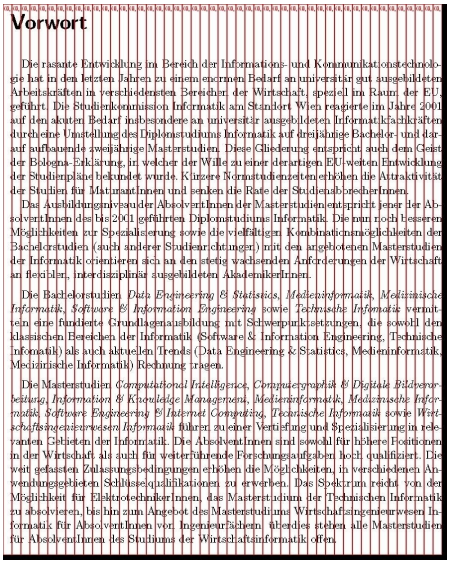
5 Pre-Processing - Up/Down Orientation Detection

We present a tractable method for one phase of the pre-processing step, namely the up/down orientation detection of shreds. The idea behind this method is

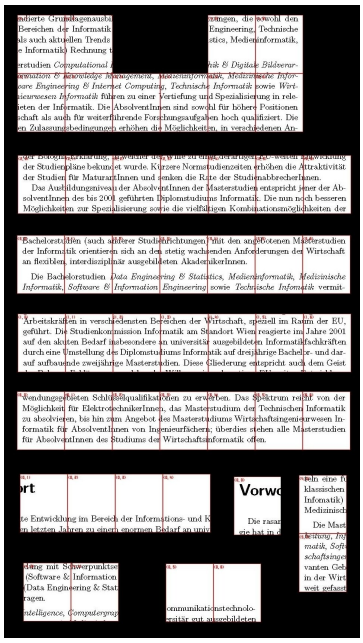
[6] A "99.95% stopping condition" means that the algorithm continues running only while it is at least 99.95% sure of the correctness of its next move.



(a) Cross-cut document, full solution.



(b) Strip-cut document, full solution.



(c) Cross-cut document, partial solution.

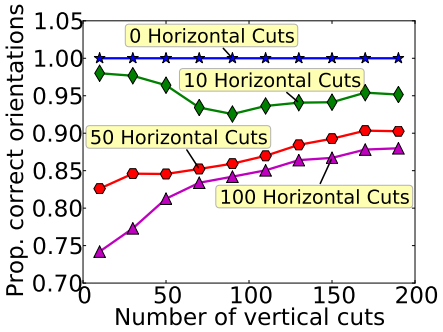
Fig. 5: Figures **a** and **b** show full reconstructions on the cross-cut variant (64% correct) and the strip-cut variant (100%correct). Figure **c** shows a partial reconstruction (with a threshold of 99.5%) which successfully reduces the search space from 49 to 10 shreds while not introducing any errors.

to find the *inner* and *outer* rows of each shred, to then count the number of black pixels in the corresponding *upper regions* and *lower regions*, and to finally predict that more black pixels will be present in the upper region (see Figure 6).

In this formulation, we define the outer rows as being delimited by those y coordinates in which a transition is made from a row of pixels that contains no black pixels to a row that contains at least 1 black pixel (or vice versa). We then filter the resulting, preliminary, outer rows such that only those taller than a minimum threshold are kept, since a row that is only 1 or 2 pixels tall is quite likely spurious. Subsequently, we select an inner row inside each outer row by

(a) The inner lines delimit the inner row, and the outer lines the outer row.

(b) For Roman script, the upper region will tend to contain more black pixels than the lower one.



(c) The proportion of correctly oriented shreds as the number of cuts grows.

Fig. 6: Figure **a** shows an example of inner and outer rows for one word and Figure **b** shows the corresponding upper and lower regions. Figure **c** shows the results obtained by using these upper and lower regions for orientation detection.

searching for the two y coordinates exhibiting the greatest difference between the sum of black pixels in consecutive rows of pixels. Thus, the greatest increase in number of black pixels between 2 consecutive rows is the upper limit of the inner row, and the greatest decrease is the lower limit of the inner row.

As can be seen in Figure 6, the results are perfect for strip-cut documents (i.e. 0 horizontal cuts) and steadily degrade as we introduce more horizontal cuts. This happens because horizontal cuts reduce the number of rows printed on each shred and therefore make the system more vulnerable to noise. One apparently odd thing about these results is that, for the lower two curves, the performance improves as the number of vertical cuts increases. This behaviour is caused by completely white pieces, which are declared as being always correctly oriented. For the lower two curves, increasing the number of vertical cuts increases the number of white pieces faster than the added noise can degrade the performance, so the overall proportion of correctly predicted orientations increases.

6 Conclusions and Future Work

This paper presents a modular and composable framework for the shredded document reconstruction problem and provides sample solutions for several of its components. Specifically, we propose: a probabilistic scoring function which outperforms currently used alternatives; a search heuristic which can either solve simpler reconstruction problems or reduce the search space for more complex ones; and an up/down orientation detection method which obtains good results on taller shreds while being very computationally efficient.

Future work will look at implementing more advanced score and search functions. Solving the cross-cut domain will likely require scoring functions which employ computer vision techniques and search function which solve the cascading

problem by performing a partial exploration of the search tree. Additionally, the current state of the pre-processing components makes it necessary to spend a significant amount of time arranging and scanning the shreds, as to minimise noise. Further work is required to create more robust and efficient pre-processing components that will lead to an efficient unshredding pipeline. Lastly the user-interaction aspect needs to be more closely analysed. Good progress has been made by the authors of [9, 13], which have shown that a fully automatic reconstruction method can be modified to incorporate user input with relative ease. The question of how to extract the maximum amount of information from a minimum number of user interactions remains, however, open.

References

1. H. Aradhye. A generic method for determining up/down orientation of text in roman and non-roman scripts. *Pattern Recognition*, 38(11):2114–2131, 2005.
2. B. Biesinger. Enhancing an evolutionary algorithm with a solution archive to reconstruct cross cut shredded text documents. Bachelor's thesis, Vienna University of Technology, Austria, 2012.
3. P. Butler, P. Chakraborty, and N. Ramakrishan. The deshredder: A visual analytic approach to reconstructing shredded documents. In *Visual Analytics Science and Technology (VAST), 2012 IEEE Conference on*, pages 113–122. IEEE, 2012.
4. R. Caprari. Algorithm for text page up/down orientation determination. *Pattern Recognition Letters*, 21(4):311–317, 2000.
5. D. Heingartner. Back together again. *New York Times*, 2003.
6. J. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society*, 7(1):48–50, 1956.
7. J. Perl, M. Diem, F. Kleber, and R. Sablatnig. Strip shredded document reconstruction using optical character recognition. In *Imaging for Crime Detection and Prevention 2011 (ICDP 2011), 4th International Conference on*, pages 1–6. IET, 2011.
8. M. Prandtstetter. Two approaches for computing lower bounds on the reconstruction of strip shredded text documents. Technical Report TR18610901, Technishe Universitat Wien, Institut fur Computergraphik und Algorithmen, 2009.
9. M. Prandtstetter and G. Raidl. Combining forces to reconstruct strip shredded text documents. In *Hybrid Metaheuristics*, pages 175–189. Springer, 2008.
10. M. Prandtstetter and G. Raidl. Meta-heuristics for reconstructing cross cut shredded text documents. In *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation*, pages 349–356. ACM, 2009.
11. C. Schauer, M. Prandtstetter, and G. Raidl. A memetic algorithm for reconstructing cross-cut shredded text documents. In *Hybrid Metaheuristics*, pages 103–117. Springer, 2010.
12. A. Skeoch. An investigation into automated shredded document reconstruction using heuristic search algorithms. Ph.D. thesis, University of Bath, UK, 2006.
13. A. Sleit, Y. Massad, and M. Musaddaq. An alternative clustering approach for reconstructing cross cut shredded text documents. *Telecommunication Systems*, pages 1–11, 2011.
14. H. Zhang, J. Lai, and M. Bacher. Hallucination: A mixed-initiative approach for efficient document reconstruction. In *Workshops at the Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.