7-1-2009

Style-Based Ballot Mark Recognition

Pingping Xiu
Lehigh University

Daniel Lopresti
Lehigh University

Henry Baird
Lehigh University

George Nagy
Rensselaer Polytechnic Institute

Elisa Barney Smith
Boise State University

 DOI: 10.1109/ICDAR.2009.273

2009 10th International Conference on Document Analysis and Recognition

Style-Based Ballot Mark Recognition

Pingping Xiu* Daniel Lopresti* Henry Baird* George Nagy† Elisa Barney Smith‡

Abstract

The push toward voting via hand-marked paper ballots has focused attention on the limitations of current optical scan systems. Discrepancies between human and machine interpretations of ballot markings can lead to a loss of trust in the election process. In this paper, a style-based approach to ballot recognition is proposed in which marks are recognized collectively rather than in isolation. The consistency of a voter's style is leveraged to improve the overall accuracy of the system. We compare style-based recognition to various kinds of singlet classifiers and show that it outperforms them by a substantial margin.

1. Introduction

Since the contentious recount in the 2000 U.S. presidential election, followed by the subsequent passage of the Help America Vote Act (HAVA) [2], a number of electronic voting systems have appeared on the market. Compared to existing approaches such as mechanical lever machines, punched cards, and hand-counted paper ballots, direct recording electronic (DRE) voting offers several advantages, including fast tabulation of election results and user interfaces designed to address the needs of disabled voters. The security of DRE's, however, has been called into serious doubt. In a famous study of the source code for the Diebold (now Premier) Accu-Vote TX, a team of researchers from Johns Hopkins identified a wide range of software vulnerabilities, many of which could compromise the results of an election. There is no reason to believe that other DRE's are immune, since any complex hardware/software system will exhibit similar flaws.

This has led to a push toward the incorporation of physical (*i.e.*, paper) records that can be verified independently of the electronic tally. Burr, *et al.* introduced the concept of *software independence* as a way of better assuring the trustworthiness of voting systems [1]. A system is software independent if a previously undetected error in its software cannot cause an undetectable change in the outcome of an election. In other words, we can trust the results without requiring assumptions about the correctness of the software.

The best-known examples in the software-independent category are optical scan (op-scan) systems that employ hand- or machine-marked paper ballots [4]. Errors in an op-scan system's software can be caught and corrected during manual audits conducted after the election with the help of the paper ballot record. As a result, op-scan voting is growing more widespread and has become an important topic to study from a variety of perspectives [3].

Reading ballots may seem like a relatively simple forms processing application; it is not. *Voter intent* is the driving definition of what constitutes a legal vote. Voting is distinctive in that it touches upon an extremely broad segment of a country's population, including citizens who are illiterate or who are aged or suffer from physical or cognitive handicaps that interfere with their ability to understand and carry-out instructions. Votes cannot be invalidated because they fail to satisfy some arbitrary predefined criterion – if a human judge would interpret a marking as a vote (or, conversely, as a non-vote), then the voting machine should do the same. Prior assumptions about voter preferences (*e.g.*, if someone votes for a candidate from Party A in one race, they are likely to vote for a candidate from the same party in a second race) must never be made. Some particularly challenging examples are on public display as a result of the recount in a disputed 2008 senatorial election in Minnesota [7]. The history of relevant document analysis research stretches back over 20 years [8].

Despite the wide range of ballot marking styles across a population, we wish to explore the assumption that individual voters will tend to be consistent. This might mean always filling in oval targets on a ballot, or always using check marks to indicate choices. Our ultimate goal is to develop mark-reading algorithms that closely replicate the human ability to discern voter intent. For this purpose, style should be exploited, not avoided. A familiar concept, this notion has already been applied with success in the field of handwriting recognition [10].

*Department of Computer Science & Engineering, Lehigh University, Bethlehem, PA 18015, USA.

†Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY 12180, USA.

‡Department of Electrical and Computer Engineering, Boise State University, Boise, ID 83725, USA.

978-0-7695-3725-2/09 $25.00 © 2009 IEEE
DOI 10.1109/ICDAR.2009.273

To make this concrete in the realm of voting where representative datasets of real ballots are hard to come by, we develop an example based on three types of ballot markings:

Intentional mark A real vote as intended by the voter. In our current studies, we assume this will be a filled target oval, a check mark, or an ex (X).

Unintentional mark Any sort of noise on the ballot that has been added accidentally in or near the target oval.

Erased mark A marking that the voter has tried to erase. While voters are instructed not to use erasers, they sometimes try to do so. The correct procedure in such cases is to "spoil" the ballot and ask for a new one.

When a field is style-consistent, algorithms can be more effective in recognizing the underlying symbols [10]. This fits our application; we can assume that it is rare for voters to mark their ballots using a mixed style. Hence, we may use Sarkar and Nagy's approach to create a style-based classifier from a set of style-specialized classifiers and thereby improve recognition accuracy.

2. Theoretical Background

The mathematics of style-based recognition are presented in [10]. As in that work, we consider fields of L isogeneous patterns represented by feature vectors $x_1, x_2, \ldots, x_L$. Each pattern, x_l, belongs to one of C classes: $c_l \in 1, 2, \ldots, C$. The goal is to deduce the class of a pattern from the observed feature vectors.

For each field, we define the field feature vector, x, as the concatenation, $x_1, x_2, \ldots, x_L$, of the constituent pattern feature vectors. The concatenation of pattern classes is called the field class or field identity, $c = (c_1, c_2, \ldots, c_L)$.

The essential aspect of a style-consistent model is the statistical dependence among pattern-features in a field. In contrast, in a singlet model, we assume that pattern-features in a field are class-conditionally independent:

$$p(x|c) = p(x_1, x_2, \ldots, x_L|c_1, c_2, \ldots, c_L) = \prod_{l=1}^{L} p(x_l|c_1, c_2, \ldots, c_L). \quad (1)$$

We can simplify further by assuming that the l^{th} pattern feature, x_l, depends on the class of the l^{th} pattern, but is independent of all of the other pattern-classes:

$$p(x|c) = \prod_{l=1}^{L} p(x_l|c_l). \quad (2)$$

When each pattern-class can be rendered in different styles, the resulting conditional pattern-feature probability is a mixture distribution. For K styles, $1, ..., K$, we have:

$$p(x_l|c_l) = \sum_{k=1}^{K} a_k p(x_l|k, c_l). \quad (3)$$

where a_k is the probability of occurrence of style k. Substituting this in Eqn. (2), the field-class conditional field-feature density is:

$$p(x|c) = \prod_{l=1}^{L} \sum_{k=1}^{K} a_k p(x_l|k, c_l). \quad (4)$$

While the above formula accounts for multiple pattern styles, it does not model the consistency of style within a field. Thus, different patterns in a field can be generated from different styles. In our style-consistent model, field-features have mixture distributions induced by styles while, within a field, all patterns come from the same style:

$$p(x|c) = \sum_{k=1}^{K} P(k|c) p(x|k, c) = \sum_{k=1}^{K} a_k p(x|k, c), \quad (5)$$

$$p(x|c) = \sum_{k=1}^{K} \alpha_k \prod_{l=1}^{L} p(x_l|c_l, k). \quad (6)$$

So the most probable field-level string is:

$$C^* = \arg\max_c \sum_{k=1}^{K} \alpha_k \prod_{l=1}^{L} p(x_l|c_l, k). \quad (7)$$

The computational cost for obtaining C^* is exponential, because for each string c we will compute Eqn. (6). If we change the sum to a max, however, and move it inside the product, the cost is lowered significantly; the problem is reduced to one where we can use a style-specific classifier to provide interpretations, then choose the most confident of these as the result:

$$C^* = \arg\max_c \prod_{l=1}^{L} \max_k (p(x_l|c_l, k)). \quad (8)$$

As was demonstrated in [10], this should not be too different from using the true formula in Eqn. (7).

3. Experimental Evaluation

3.1. Test data

Before classification, ground-truthed training samples must be prepared. We hand-marked a set of ballot images, aligning them with the blank ballot. The alignment procedure first pinpoints pairs of registration points between the

Table 1. Test set sizes (training sets are 5×).

Classifier	Votes	No-Votes (noise, blank)
Check	140	180
Ex	140	180
Filled	154	235
Blend	434	295
Separate	140 check + 140 ex + 154 filled	295

two images to be aligned, then calculates three variables: a two-dimensional shift vector, a scaling factor, and a slant angle. This procedure is implemented in PERL.

Next, we use the BallotGen toolkit developed for the PERFECT project [6] to generate the ground-truthed mark images for training. BallotGen is written in Tck/Tk, and additional support was added for introducing small variations to amplify the training set, such as shift, rotation, etc. The standard netpbm library is used for manipulating images. Target bounding boxes are all the same size, 155×105.

Input patterns are to be classified as either *vote* or *no-vote*. Samples from the three marking styles we consider in this paper are shown in Figure 1, while the characteristics of our dataset are listed in Table 1.

3.2. Classifier technology

We use Modified Quadratic Discriminant Functions (MQDF) as our classifier technology [5]. The probability distributions for MQDF have a large variance because it operates in a high dimensional space. Hence, it is appropriate to adopt the approximation version given in Eqn. (8), as opposed to the exact formulation of Eqn. (7).

As our feature set, we use polar transformations and the magnitude portion of the 2-D Fast Fourier Transform, yielding a total of 300 features. Our rationale is based on the observation that the markings we wish to recognize are usually symmetric. A blank oval's 2-D FFT concentrates its energy in certain frequencies, while pepper noise is spread out. Checks and ex marks concentrate their energies differently than ovals. Finally, filled ovals have significantly higher DC components than the other types of marks. Hence, this appears to be a reasonable feature set for our test case.

To characterize the problem, we trained an MQDF classifier on four classes: *check*, *ex*, and *filled* marks, as well as *no-vote* instances (blank targets and partially-erased marks). We then tested the classifier on the training set to yield the confusion matrix shown in Table 2. From this, we can see that no-votes and filled marks are rarely confused. However, noisy no-votes can be confused with checks and ex's.

Table 2. Confusion matrix for the classes.

Class	No-Vote	Check	Ex	Filled
No-Vote	71.33%	16.00%	11.67%	01.00%
Check	32.14%	34.29%	33.57%	00.00%
Ex	00.71%	11.43%	85.00%	02.86%
Filled	00.00%	00.00%	04.29%	95.71%

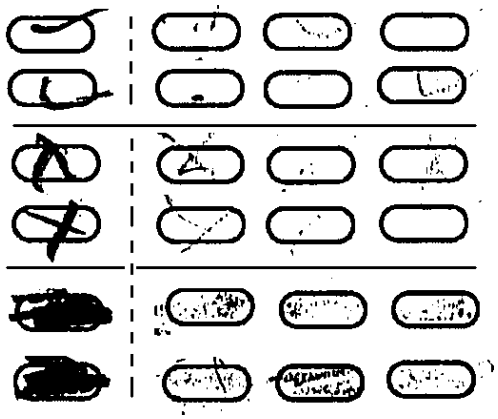

Figure 1. Three different marking styles: check, ex, and filled, and their corresponding noisy (no-vote) patterns. Noisy inputs are generated by attempting to erase a mark, something that voters occasionally do.

If we know which style we are recognizing (check, ex, or filled), we can use a specialized recognizer and presumably achieve higher accuracy than a more general-purpose classifier trained on all styles. In our experiments, we train a Check recognizer using checks as the vote class, with erased checks, blank targets, and pencil-drop noise as the no-vote class. Ex and Filled recognizers are trained likewise.

In designing general-purpose classifiers to recognize inputs in any of the three marking styles, we employ two strategies. The first is to train a two-class classifier: vote and no-vote. Marks from all styles are treated as representing a single vote class, and no-vote patterns (including blank targets and erased marks) are treated as a single no-vote class. It may, of course, be problematic to mix distinct mark types in a single class. Another way to design a classifier is to treat the marks from each style as separate classes, while regarding all no-votes as equivalent. This yields a four-class classifier: check, ex, filled, and no-vote. We refer to the former as Blend, and the latter Separate.

Note that style-based recognition requires the classifiers to operate on the same feature space because their posterior probabilities will appear in the same formula and hence must derive from the same probability space. This differs

Table 3. Target-level error rates (top) and field-level error rates (bottom).

	Classifier					
Sample Set	Check	Ex	Filled	Blend	Separate	Style-based
Check	2.36%	7.46%	25.00%	1.97%	4.35%	2.78%
Ex	0.40%	0.34%	16.16%	0.40%	0.40%	0.35%
Filled	2.75%	2.38%	1.10%	2.75%	2.50%	1.09%
Average	1.84%	3.39%	14.09%	1.70%	2.42%	1.41%

	Classifier					
Sample Set	Check	Ex	Filled	Blend	Separate	Style-based
Check	38.30%	83.25%	100.00%	33.43%	61.08%	42.85%
Ex	7.77%	6.70%	99.30%	7.77%	7.77%	6.75%
Filled	53.18%	46.07%	20.75%	53.18%	48.55%	20.63%
Average	33.08%	45.34%	73.35%	31.46%	39.13%	23.41%

Figure 2. Example of a recognition result: the race's ground-truth is VNNN (Vote, No-Vote, No-Vote, No-Vote). The native Ex recognizer reads it correctly as VNNN, the Check recognizer reads it as VVNN, the Filled recognizer reads it as NNNN, the Blend recognizer reads it as VVNN, and the Separate recognizer reads it as VVNN. The style-based recognizer reads it correctly as VNNN.

from typical classifier combination strategies which employ different classifiers and which do not operate on fields.

3.3. Results

In conducting our experiments, we have five singlet classifiers at our disposal: Check, Ex, Filled, Blend, and Separate. For each class we have $1,800$ image samples. The classifiers were trained on $5/6^{th}$ of the samples, and then tested on the remaining $1/6^{th}$. We will contrast their performance with that of a style-based recognizer.

An example of a field containing four patterns is shown in Figure 2. We have two basic options for quantifying recognition rates: comparing performance at the target level or at the field level. The former measures the percentage of correct decisions for each individual vote, while the latter can be thought of as measuring the percentage of ballot races that are correctly interpreted as a whole. Only if all of the targets constituting a race (field) are recognized correctly do we consider the field correct.

We ran the five singlet classifiers and then the style-based recognizer. The field length was 20, with five of the patterns corresponding to real votes, one to an erasure, and the remaining 14 blank. The top half of Table 3 shows the error rates for individual target recognition, while the bottom half gives the field-level error rates. We note that the singlet classifiers generally do well in their own domains of expertise, but sometimes break for other kinds of inputs. The field-level error rates are higher than the target-level rates because just one error in a field renders the result incorrect.

The Blend and Separate recognizers perform similarly, with the former somewhat better than the latter. This suggests that separating the mark classes does not help. Note also that both usually beat the singlet classifiers (except in the case of filled marks). Hence, using a general-purpose classifier is better than using a wrongly-applied style-specific classifier. The style-based recognizer is rarely the best for a specific kind of input, but its overall performance is better by a substantial margin. This is, in fact, our goal. While the quoted error rates are not low enough for commercial election equipment, this is still compelling evidence of the promise of style-based ballot mark recognition.

Class-specific error rates for the different classifier schemes are displayed in Table 4 (the error rate for recognizing blank targets is 0.0% across the board). Again, the style-based recognizer is not uniformly the best. The Blend classifier has the lowest error rate for recognizing votes (2.92%), whereas the Filled classifier has the lowest error rate for recognizing noise that should be considered a no-vote (4.75%). However, Blend does a poor job with the noise that arises when filled marks are erased (51.35% error rate), and Filled does badly for votes that are marked as a check (100.00%) or an ex (64.63%), which it likely regards as noise. The style-based approach yields the most broadly competent classifier in the group, with a 4.32% error rate at recognizing votes of all types, and a 6.52% error rate at recognizing noise as no-vote.

Table 4. Class-specific mark error rates (top) and noise error rates (bottom).

	Classifier					
Sample Set	Check	Ex	Filled	Blend	Separate	Style-based
Check	8.94%	29.49%	100.00%	7.14%	17.06%	10.61%
Ex	0.91%	0.91%	64.63%	0.91%	0.91%	0.91%
Filled	0.71%	0.71%	1.53%	0.71%	0.71%	1.43%
Average	3.52%	10.37%	55.39%	2.92%	6.23%	4.32%

	Classifier					
Sample Set	Check	Ex	Filled	Blend	Separate	Style-based
Check	2.58%	1.68%	0.00%	3.67%	1.68%	2.48%
Ex	3.43%	2.35%	0.00%	3.43%	3.43%	2.40%
Filled	51.35%	43.97%	14.25%	51.35%	46.50%	14.68%
Average	19.12%	16.00%	4.75%	19.48%	17.20%	6.52%

4. Conclusions

Style-based ballot mark recognition exploits voter consistency to boost accuracy. In this paper, we tested the technique on a focused dataset, but we believe that its potential for improving the processing of hand-marked paper ballots is enormous. The result should be a better realization of "voter intent," the legal definition of what constitutes a vote.

The error rates we report are relatively high because the testing samples are intended to be hard. Moreover, our current implementation does not include the common sorts of pre-processing steps one would find in a more complete implementation. We plan to examine different classifier strategies and feature sets to raise mark recognition to an acceptable level for voting applications.

From Figure 1, we can see that noise has style, too. One person's attempt to cancel a vote may look quite similar to another person's attempt to record a vote. In this case, style-based recognition may perform even better, for it excels at disambiguating conflicting classes of different styles. We plan to collect data to confirm this hypothesis.

We may also try varying the field length to determine if there is an optimal value. As we have noted, there appears to be no need to pay the exponential computation time penalty. Finally, we conclude by encouraging the document analysis community to consider ballot reading as a worthy research topic with tremendous social importance. The PERFECT project has as its goal the development of more accurate mark recognition algorithms for op-scan systems [6, 9].

5. Acknowledgments

This work was supported in part by the National Science Foundation under award numbers NSF-0716368, NSF-0716393, NSF-0716647, and NSF-0716543. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the National Science Foundation.

The MQDF classifier we used was the result of Pingping Xiu's internship at Google, Inc. during the summer of 2008.

References

[1] W. Burr, J. Kelsey, R. Peralta, and J. Wack. Requiring software independence in VVSG 2007: STS recommendations for the TGDC, November 2006. http://www.voteraction.org/files/Soft_Ind_VVSG2007-20061120.pdf.

[2] Help America Vote Act of 2002. http://www.fec.gov/hava/hava.htm.

[3] P. S. Herrnson, R. G. Niemi, M. J. Hanmer, B. B. Bederson, F. C. Conrad, and M. W. Traugott. *Voting Technology: The Not-So-Simple Act of Casting a Ballot*. Brookings Institution Press, 2008.

[4] D. W. Jones. Counting mark-sense ballots, February 2002. http://www.cs.uiowa.edu/ jones/voting/optical/.

[5] C.-L. Liu, H. Sako, and H. Fujisawa. Discriminative learning quadratic discriminant function for handwriting recognition. *Institute of Electronics, Information, and Communication Engineers*, 15(2):430–444, March 2004.

[6] D. Lopresti, G. Nagy, and E. B. Smith. A document analysis system for supporting electronic voting research. In *Proceedings of the Eighth IAPR Workshop on Document Analysis Systems*, pages 167–174, Nara, Japan, September 2008.

[7] (Minnesota) challenged ballots: You be the judge, January 2009. http://minnesota.publicradio.org/features/2008/11/19_challenged_ballots/.

[8] G. Nagy. Twenty years of document image analysis in PAMI. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):38–62, 2000.

[9] Paper and Electronic Records for Elections: Cultivating Trust (PERFECT), 2009. http://perfect.cse.lehigh.edu/.

[10] P. Sarkar and G. Nagy. Style consistent classification of isogenous patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(1), January 2005.