Document Analysis Techniques for Automatic Electoral Document Processing: A Survey

J. Ignacio Toledo[1], Jordi Cucurull[1], Jordi Puiggalí[1], Alicia Fornés[2], and Josep Lladós[2]

[1] Scytl Secure Electronic Voting, Barcelona, Spain,
{JuanIgnacio.Toledo,Jordi.Cucurull,Jordi.Puiggali}@scytl.com
[2] Computer Vision Center, Universitat Autònoma de Barcelona, Barcelona,Spain,
{afornes,josep}@cvc.uab.es

Abstract. In this paper, we will discuss the most common challenges in electoral document processing and study the different solutions from the document analysis community that can be applied in each case. We will cover Optical Mark Recognition techniques to detect voter selections in the Australian Ballot, handwritten number recognition for preferential elections and handwriting recognition for write-in areas. We will also propose some particular adjustments that can be made to those general techniques in the specific context of electoral documents.

Keywords: Document Image Analysis, Computer Vision, Paper Ballots, Paper Based Elections, Optical Scan, Tally

1 Introduction

While remote or poll-site electronic voting is gaining more and more acceptance worldwide, many elections are still paper based. Be it for tradition, for its simplicity, because it leaves a physical evidence of the vote or because of a restrictive electoral law, there are several countries that are not willing to abandon paper based elections yet. However, this does not mean that they are not willing to use modern technology in elections.

Countries with complex electoral systems, like the US, have been exploring how to automate the tally for paper based elections for decades. Mark sense scanners, first developed for educational testing, have been used for ballot processing since the 1950's. They were based on a ballot printed with a special ink, that was invisible to the sensor, and the use of index marks to define the position of the voting targets. In the 1990's, devices using imaging technology were developed. They used fiducial marks that allowed the scanner to interpolate the voting targets and counted the number of dark pixels in each area. More recently, in 2006, a patent was granted to a device based on edge detection, which could detect empty voting targets (ovals) and filled voting targets.

We can see a trend moving from solutions requiring specific hardware to more generic hardware-independent solutions using computer vision techniques.

However, there are still a lot of challenges to be able to support more complex elections. In the document analysis field, techniques have been developed to process different kind of documents. To our knowledge, the work specifically applied to electoral documents has mainly dealt with Optical Mark Recognition. In this paper we will discuss several of the techniques developed in the document analysis community that can be applied in the electoral document context, while pointing out some improvements that can be done using knowledge of the electoral process.

This paper is organized as follows: In Section 2 we will discuss the more relevant preprocessing steps applied in document image analysis. In Section 3 we will deal with ballots, starting with the most common issue, detecting filled voting targets. We will also deal with preferential voting, where voters have to sort candidates according to their preferences by assigning them a number, and the hardest problem we can find in ballots, write-ins, where voters can write in the name of the candidate in a designated area will be discussed. In Section 4, we will see how we can apply most of the techniques previously discussed in another kind of electoral document, the ballot statement. In Section 5 we show a few small security enhancements that can be easily implemented. Finally we draw some conclusions and outline some possible lines of future work.

2 Preprocessing

In image processing, before trying to understand a document image, we can try to simplify the problem by removing some sources of variance. The same intensity value can sometimes represent a black pixel or white (background pixel) depending on the acquisition device. It is also very common to find different skews on each scan, due to small misalignments when feeding the paper sheet into the scanner. Finally the image can be noisy. We will discuss techniques to address each of these problems.

A key preprocessing step in most document analysis tasks is image binarization. That is, determining if a pixel of the image should be considered "black/foreground" or "white/background" depending on whether its darker or brighter than a certain threshold value. If the image acquisition is done in a very controlled environment, a global threshold value can be predefined. This is the less flexible approach and it can fail if you have to use scanners from different manufacturers or with different contrast response. There are also several different methods to automatically find optimum global thresholds. One of the most widely used method is Otsu's method [17]. This method is based on iterating through the 256 possible threshold values of a typical 8-bit gray level image finding the value that minimizes the intra-class variance (which is equivalent to maximizing the inter-class variance). This kind of methods would allow us more flexibility in the requirements of a particular scanner configuration.

There are also adaptive threshold methods like Niblack [16], Bernsen [4] or Sauvola [21]. In this kind of methods, instead of selecting a single threshold value for the whole image, the threshold value is determined for each individual pixel,

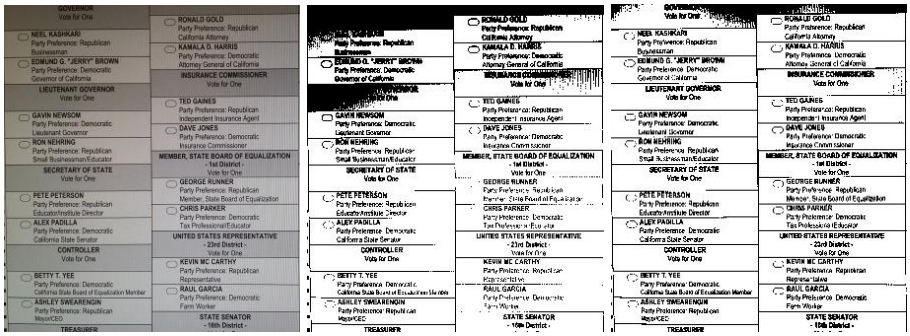


Fig. 1. The original ballot image acquired with a camera (left). The image thresholded with Otsu's Method (center) and with Sauvola's Method (right). We can see how using Otsu's method the darker areas of the ballot become black while voting targets in the lighter areas disappear, showing the limitations of setting a global threshold.

taking into account its neighbors in a local area of a predefined size. In the case of Sauvola, a widely used method for documents, the mean and the standard deviation in the local area are calculated. Then each pixel is classified as dark, if it is at least k times (a parameter) the standard deviation darker than the mean in that area. This kind of binarization methods are specially interesting if there are illumination changes (as for instance when the ballot images are acquired using a camera), noise in the image, or stains or folding marks in the ballot. A very interesting survey on both global and local thresholding algorithms can be found in [23, 19] showing that despite being a mature research area, there is still interest in the community for binarization techniques. See Fig. 1.

Another key preprocessing step is the removal of the skew; there are also several approaches to do this. One of the most common approaches [13], is based on rotating the document in all allowed skews (i.e, from -10 to 10 degrees with a precision of 1 degree), trying to find the right orientation. There are several ways to find out the correct orientation. Assuming an horizontal writing, the document will have the correct orientation when the horizontal projection histogram has a higher variance. Also, if there is a long horizontal line separating two areas, the correct orientation would be the one that produces the highest peak value for a specific line in the horizontal projection histogram. Another common approach would be to use the Hough Transform [9, 3, 24]. Using the Hough Transform we get the equation of all the lines $y = ax+b$ that we can find in a document, making it trivial to find the skew of the document. Nevertheless, mainly because of the computational cost of the Hough Transform, methods based on the horizontal projection are more commonly used.

In some cases, after thresholding and skew correction, some noise removing algorithms can be applied. For instance, the median filter can be useful to remove "salt and pepper" noise (isolated black or white pixels). Mathematical morphology operators [22] (opening, closing, erosion, etc.) can also be used in

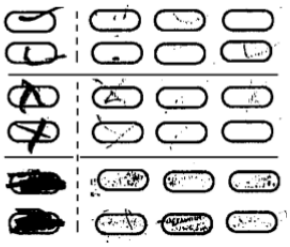


Fig. 2. Three different marking styles: check, ex, and filled, and their corresponding noisy inputs generated by the voter attempting to erase a mark. Extracted from [29].

case we need to remove artifacts with a specific shape/size or connect some broken shapes.

3 Ballots

The most common election document is the ballot. Ballot design can have a high variability depending on the electoral system of each country or state. The kind of challenges we can find in ballots can be divided in three big groups: mark recognition, preferential voting and write-ins. We will review each of them in the following subsections.

3.1 Mark Recognition

The ballots used in most of the elections consist of a grid where a voter selects k out of n candidates for each contest by filling in empty voting targets in predefined locations. In the most simple case, there will only be one ballot model. In this case, the recognition software will only require a mapping from a filled voting target (dark pixels in a certain area) to a candidate name.

However, in most complex elections we usually have to deal with different ballot models (in different languages, or different districts with different contests). The first step that the software will perform (after the preprocessing step) is to identify the ballot model. The most popular solutions use QR-codes or barcodes to identify each model. After reading the barcode and identifying the ballot model, the configuration for that particular model can be loaded, that is, the position of the pixels of each voting target and the candidate it is associated to. If there are enough dark pixels in that area, it means that the voter cast a vote for that candidate. This kind of approach looks very efficient and simple, but it has problems because some voters do not fill the voting target completely or place their mark near but not inside the voting target. What kind of marks are considered a valid vote depends on the electoral law, and traditional approaches like this are lacking in flexibility.

An alternative approach could allow us to perform both the ballot model and mark detection at the same time, avoiding the need of barcodes. To do that,

we need a template image of an empty ballot of each ballot model. The process would consist in computing the difference of the ballot (after preprocessing) and each of the templates. The actual ballot model will have the smaller difference, and that difference would be the marks made by the voter. This difference will usually have an amount of noise due to small misalignments, dust or different scanning conditions. To obtain a mark detector that is less sensitive to noise, several approaches are discussed in [25, 27], like using a distance transform to detect safe and unsafe zones, depending on their distance to black pixels, using Gaussian filters to smooth the images before performing the subtraction or using morphological filters. Some authors try to detect a grid for possible positions of marks by analyzing the geometry of the ballot [26]. Other authors simply require user collaboration to tag a blank voting target and locate the rest using pattern matching techniques. Once they know where voting targets are, they search for filled in targets in that region [28, 12].

One drawback of the approaches described above is that they mainly rely on the size of the mark. Usually, some voters do not follow exactly the instructions to completely fill the voting target area, and use marks like X or ✓(see Fig. 2). Since most electoral laws define a vote in terms of voter intent, we have to be able to detect these marks. A possibility suggested in [29], assuming that the voter makes consistent marks, is to train classifiers taking into account the style of the marks, improving mark detection.

3.2 Preferential Voting

In some elections the voter is allowed to perform preferential voting. In that scenario detecting a mark in a voting target is not enough. In preferential voting, the voter assigns a number to each candidate indicating their preference, so we have to classify the marks we find as belonging to a particular class (i.e. "1", "2", etc.).

The problem of identifying the particular class of an image among a possible set of classes is one of the big challenges in computer vision. Fortunately, in handwritten numbers, the number of different classes is small (only ten different classes) and there have been free datasets available for years. The main challenge is the huge difference in writing styles. Classifying handwritten isolated digits has been tackled by computer vision for the last three decades and there is now a wide variety of techniques that allow us to perform the recognition of individual digits with less than a 2% error rate [15] on the popular MNIST dataset [14]. See Fig. 3 for some examples.

Recently, a multicolumn convolutional deep neural network trained for weeks with several GPU has surpassed human performance in this task, achieving an error rate of 0.23% [5]. Convolutional neural networks combine the ability to learn low level features (convolutional layers) with the invariance to traslation and scale given by max-pooling layers. Deep neural networks try to emulate the hierarchical representations of the human brain, where the first layers learn low level features, and the layers above learn higher level features (non-linear

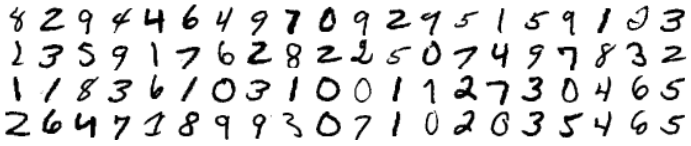


Fig. 3. Some examples from the MNIST dataset. It's a common benchmark for isolated handwritten digit recognition consisting of 60,000 digit images from approximately 250 different writers.

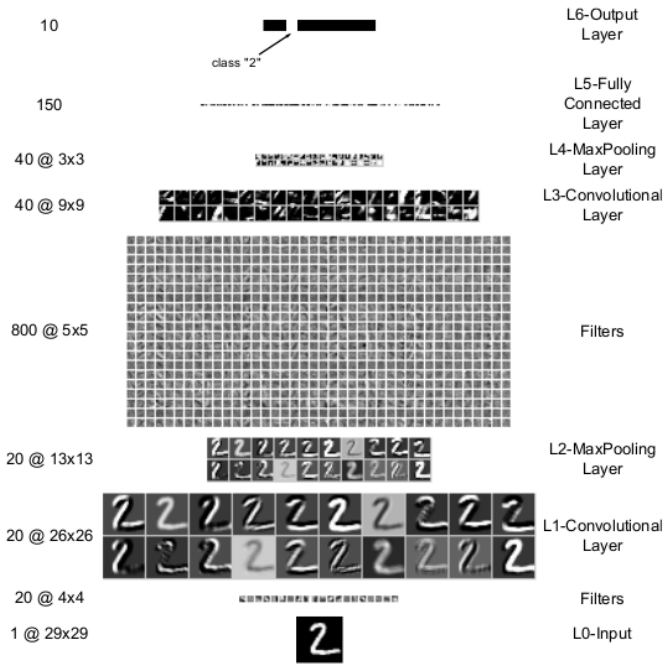


Fig. 4. The arquitecture of one column of the convolutional neural network that achieved the best scores so far in handwritten digit recognition on the MNIST dataset. The response for each neuron to the input image is also shown as an image. Extracted from [5]

combination of the low level ones). The last layer is the actual classifier (a non-linear multiclass logistic regression) that outputs the probability of each class given that particular input image.

In practice, these "deep learning" systems are still difficult to train because they require long training times, huge amounts of data, careful tuning of network parameters, and expertise in GPU programing. For that reason, traditional systems using handcrafted features like Histogram of Oriented Gradients (HOG) [11] and classifiers like Support Vector Machines (SVM) are still a very popular approach [6]. SVM also output the probability of the observation be-

longing to each specific class. This is very important because it gives us not only a most probable label, but also a confidence on that prediction.

In electoral documents there is additional context information that can be used to further reduce the error rate. Usually a number cannot be repeated within the same contest (there cannot be two candidates with the same preference in the same contest) and usually they have to be correlative (i.e a voter cannot assign a preference "3" without previously assigning preferences "1", and "2"). Instead of individual classifications, we are facing a problem of a set of observations with some restrictions that can help us lower our error rate even more. Finally, the number of preferences a single voter can choose is probably less than ten, that would reduce the number of classes (which has a great impact in error rates). For example, usually the digit 1 is mistaken by a 7, or the digit 3 with a 5 or an 8, so if we have less than 7 preferences to assign, the error rate would drastically decrease. Finally, since these techniques also provide a confidence level on the classification result, this confidence level can be used to discard an ambiguous ballot and ask for a human decision if the confidence is below a certain threshold. This approach of combining Intelligent Character Recognition techinques with human inspection of dubious ballots has been used successfully in several elections in the Australian Capital Territory [2].

3.3 Write-In

Recognizing the text in write-in areas is the most difficult problem we can find in electoral documents. Handwriting recognition can be performed with online or offline information. In online systems, the tempral sequence of the handwritting is available whereas in offline scenarios, we only have an scanned image available. While the recognition rate is better in the online scenario, we discarded its usage in our systems because: 1) it requires special hardware (a digital pen or digitizing board that records the (x,y) position of the pentip at each timestep) and 2) it has security implications because it detaches the voter input from the ballot background, forcing to perform audits on the physical ballots to avoid ballot tampering.

Offline cursive handwriting recognition, with open vocabularies in a multi-writer scenario is still an open problem, the state of the art [8] character error rates are around 18-19%. Probably the main reason that can explain why this is a hard problem is the so called 'Sayre's Paradox'. This paradox states that handwriting recognition is a "chicken-egg" problem. In order to segment a cursive word into characters you need to recognize the characters first, but to recognize the characters you first need to segment them out. A way to circumvent this problem is to use segmentation-free techniques. Also we have to keep in mind that cursive handwriting has huge variability, thus most of the approaches include a preprocessing step trying to normalize the slant, horizontal and vertical size of the characters and, in some cases, even the stroke width.

The key idea is to model the handwritten text line or word like a temporal series of observations with a "sliding window approach". See the example of the sliding window approach on a previously normalized handwritten text line in

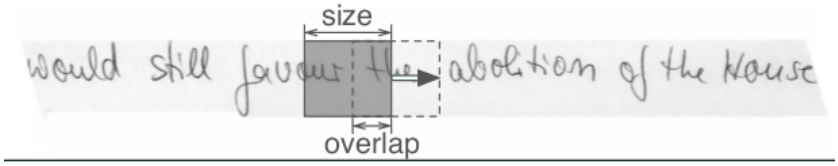


Fig. 5. The "sliding window". Extracted from [7]

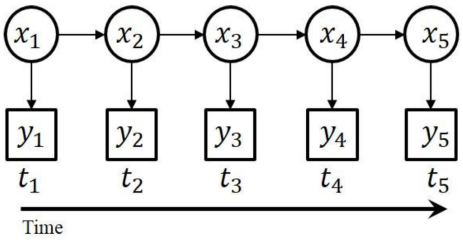


Fig. 6. In Hidden Markov Models, the data is modeled as a series of observations generated by a hidden state that is only dependent on the state at the previous time step.

Fig. 5. That is, we focus our attention only in a column of a few pixels wide at a time and extract some representative features in that window. There are different set of features that are used in the literature, like statistical moments, the slope of the upper and lower contour, image derivatives, the number of black and white transitions, etc. Once we have the handwritten text represented as a series of features, the correct alignment with the ground truth character sequence has to be found. Since the character sequence and the feature sequence have different lengths the alignment is not trivial.

Since the 90s, technologies like Hidden Markov Models [20, 7] have been used to address this problem [18]. Hidden Markov Models are generative models that have been adapted from the speech recognition area. According to this model, each observation(x_t) in every timestep is conditionally dependent only from a latent unobserved variable (hidden state x_t), which in turn depends only on the hidden state of the previous timestep (Markov process). Given a number of states (x), a matrix T of allowed transitions among them $p(x_t|x_{t-1})$, and a parametric probability distribution P for $p(y|x)$, the Baum-Welch algorithm can be used to train the system, that is, finding the parameters for T and P that better fit our observations. A graphical representation of the HMM can bee seen on Fig. 6

In 2009 a new algorithm was developed that allows us to use neural networks for segmentation free handwriting recognition. The algorithm, called Connectionist Temporal Classification (CTC) allows us to align two sequences of different lengths and return a differentiable error for each timestep. With the output from the CTC algorithm, and using the traditional backpropagation algorithm, it is possible to train a recurrent neural network to map the image feature rep-

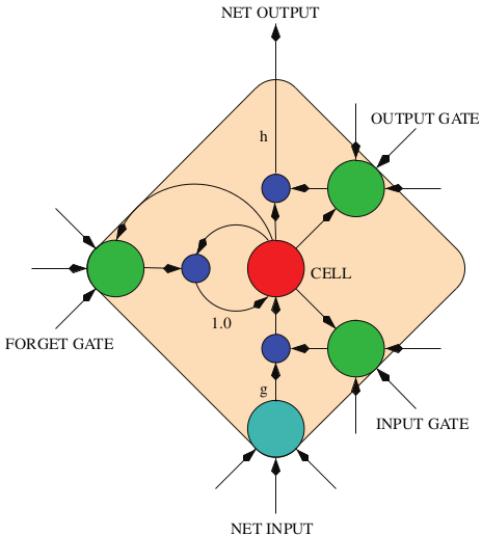
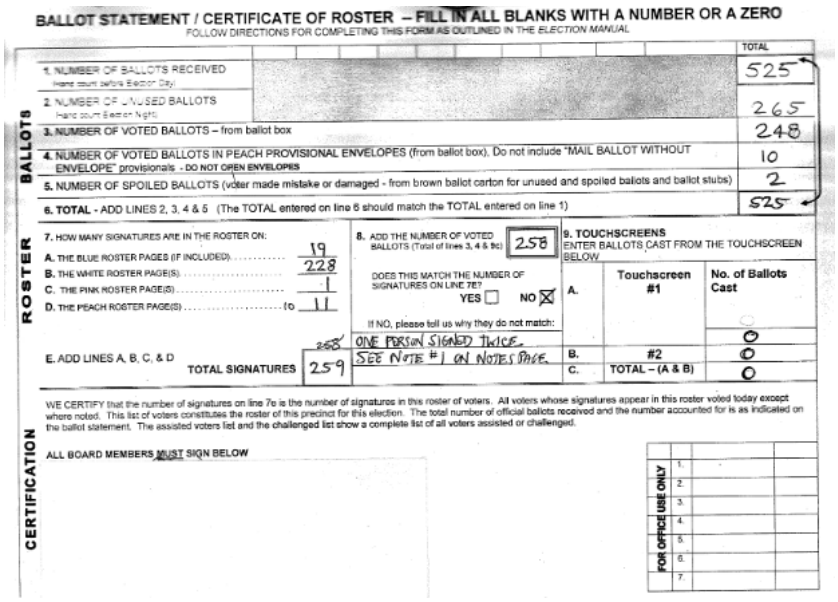


Fig. 7. A Long Short Term Memory Cell with multiplicative input, output and forget gates. Extracted from [8]

resentation with the character sequence. However, traditional recurrent neural networks have problems learning long sequences, because of a problem known as the vanishing gradient. After several timesteps, because the activation function of each neuron is smaller than 1, the error gradient fades into the network, making it unable to learn long range dependencies. This problem can be solved with the Long Short-Term Memory (LSTM) cells (Fig. 7), that incorporate input, output and forget gates, that the cell can learn to open or close depending on the input and the current state, thus allowing the network to learn arbitrarily long sequences.

The easier way to dramatically improve the recognition rate would be to change the write-in areas so that they are expected to be filled with a set of isolated capital letters. Also, in electoral documents we can assume that the content of the write-in area will be a name. We can then use a reduced vocabulary, consisting of the 5,000 most common surnames in that country, to improve the accuracy of the system in both the original connected handwriting and isolated character recognition scenarios. Finally, since the number of voters who actually use the write-ins area is usually low, there is also the option to simply detect the presence of write-in text, and mark the ballot for human inspection. This approach would still be better than current optical scan technologies, since they require the voter to fill in a mark associated to the write-in in order to process it. Requiring to fill-in that mark does not seem intuitive since, according to a study performed by Ji [10], a 49% of the voters who used write-ins forgot to fill the associated mark.


BALLOT STATEMENT / CERTIFICATE OF ROSTER – FILL IN ALL BLANKS WITH A NUMBER OR A ZERO

FOLLOW DIRECTIONS FOR COMPLETING THIS FORM AS OUTLINED IN THE ELECTION MANUAL

BALLOTS

	TOTAL
1. NUMBER OF BALLOTS RECEIVED (Hand count before Election Day)	525
2. NUMBER OF UNUSED BALLOTS (Hand count Election Night)	265
3. NUMBER OF VOTED BALLOTS – from ballot box	248
4. NUMBER OF VOTED BALLOTS IN PEACH PROVISIONAL ENVELOPES (from ballot box). Do not include "MAIL BALLOT WITHOUT ENVELOPE" provisionals - DO NOT OPEN ENVELOPES	10
5. NUMBER OF SPOILED BALLOTS (voter made mistake or damaged - from brown ballot carton for unused and spoiled ballots and ballot stubs)	2
6. TOTAL - ADD LINES 2, 3, 4 & 5 (The TOTAL entered on line 6 should match the TOTAL entered on line 1)	525

ROSTER

7. HOW MANY SIGNATURES ARE IN THE ROSTER ON:

A. THE BLUE ROSTER PAGES (IF INCLUDED) 19

B. THE WHITE ROSTER PAGE(S) 228

C. THE PINK ROSTER PAGE(S) 1

D. THE PEACH ROSTER PAGE(S) ..... (o 11

E. ADD LINES A, B, C, & D TOTAL SIGNATURES 259 (258)

8. ADD THE NUMBER OF VOTED BALLOTS (Total of lines 3, 4 & 5c) 258

DOES THIS MATCH THE NUMBER OF SIGNATURES ON LINE 7E? YES ☐ NO ☒

If NO, please tell us why they do not match:

ONE PERSON SIGNED TWICE.

SEE NOTE #1 ON NOTES PAGE.

9. TOUCHSCREENS ENTER BALLOTS CAST FROM THE TOUCHSCREEN BELOW

	Touchscreen #1	No. of Ballots Cast
A.		0
B.	#2	0
C.	TOTAL – (A & B)	0

CERTIFICATION

WE CERTIFY that the number of signatures on line 7e is the number of signatures in this roster of voters. All voters whose signatures appear in this roster voted today except where noted. This list of voters constitutes the roster of this precinct for this election. The total number of official ballots received and the number accounted for is as indicated on the ballot statement. The assisted voters list and the challenged list show a complete list of all voters assisted or challenged.

ALL BOARD MEMBERS MUST SIGN BELOW

FOR OFFICE USE ONLY

1.
2.
3.
4.
5.
6.
7.

Fig. 8. Example of a ballot statement. Extracted from [1]

4 Ballot Statements

In some elections, with very simple ballot designs (e.g Partisan Ballot), processing the ballot is extremely easy, you just have to identify the party corresponding to each ballot. In that case, human tally at precinct level is feasible. After performing the tally, the electoral officials have to fill in a report or 'ballot statement' with the election results for that precinct. We can see an example of such a document in Fig. 8.

These ballot statements usually contain handwritten numbers that represent the number of votes for a specific party, the number of eligible voters, etc. The same techniques as the one described above for "Preferential Voting" in ballots can be used. In the case of ballot statements, some integrity checks could also be performed when recognizing the digits that can help to reduce even more the classification errors (o even help to detect election official errors). Spatial grammars can be defined for a ballot statement document, that is, numbers recognized in a certain area must meet some requirements. For instance, the sum of the recognized votes of all the parties and blank votes must match the number recognized as total votes cast, which in turn has to match the number recognized as number of voters, which has to be smaller than the number of eligible voters, etc.

Some ballot statements can also contain connected handwriting. Usually the numbers are also written in text form (like the courtesy amount in cheques). We can recognize this text with high accuracy because of the very restricted

vocabulary and syntax. Since recognizing the text "thirty four" and the number '34' use different techniques to analyze different data, they can be considered independent probabilities, which can be easily combined to boost the confidence of the recognition.

To finish, usually, there is also an "observations" field, where the election officers can write free text to explain some anomaly during the election. As we explained above, unconstrained offline handwriting recognition is still an open problem. Since that field is usually empty, simply detecting if there are any observations, and asking a human operator for a transcription seems the best option.

5 Security

With traditional mark detection scanners, it may be feasible to tamper elections by replacing the original ballots by ballots with candidates in permuted order, easily affecting the election results, or by changing the configuration of the machine. Also, mark sense scanners can't detect any kind of identification mark, thus allowing vote-coercion and vote buying by corrupt election officials. If coercion or vote-buying are a concern for that particular election, we could detect possible identification marks anywhere on the ballot using document analysis techniques like the ones described above, based on the difference with a template ballot image. These same techniques would be able to detect a false ballot. In the case that we wanted to go even further (or if for some reason we can't have an image of an empty ballot for every ballot model) we could perform an OCR on the printed text to further validate the authenticity of the ballot. Knowing the font used in the ballot, the OCR can be done with almost perfect accuracy. These techniques cannot prevent pattern based marking or deliberate mismarking as way of identification. The document analysis community has also worked on signature verification. Given enough training samples from the election officers, we could verify that the signatures in a ballot statement are not forgeries.

6 Conclusions

We have reviewed the most relevant document analysis techniques that can be applied to ballot processing in all of the possible configurations: mark detection, preferential voting and write-ins. We also proposed some small improvements to the general techniques that can be applied in the specific context of ballots, like using the fact that numbers cannot be repeated in a same contest in preferential voting or that there may be less than ten different classes of numbers to improve the accuracy, or in write-ins, where we can use a vocabulary of common surnames. Using the techniques described in this paper we can go beyond the traditional Optical Mark Recognition systems and support most complex election types, dramatically reducing the need of human intervention. Finally, we

have also shown that advanced techniques for mark detection also lead to improved security. As a possible future line of research, we would like to study if it would be possible to develop a system that, by using layout detection, OCR and prior knowledge of electoral processes, could be able to interpret most common ballot designs without requiring a manual configuration for each ballot model.

7 Acknowledgements

We thank the reviewers for their suggestions and comments. This work has been partially supported by the Spanish project TIN2012-37475-C02-02 and the European project ERC-2010-AdG-20100407-269796 and by the Secretaria d'Universitats i Recerca del Departament d'Economia i Coneixement de la Generalitat de Catalunya.

References

1. Citizen's oversight projects. www.copswiki.org/Cops/BallotStatements (2009)
2. Elections ACT: Scanning of ballot papers. http://www.elections.act.gov.au /elections_and_voting/scanning_of_ballot_papers (2015)
3. Amin, A., Fischer, S.: A document skew detection method using the hough transform. Pattern Analysis & Applications 3(3), 243–253 (2000)
4. Bernsen, J.: Dynamic thresholding of grey-level images. In: International Conference on Pattern Recognition (ICPR). pp. 1251–1255 (1986)
5. Ciresan, D.C., Meier, U., Schmidhuber, J.: Multi-column deep neural networks for image classification. In: IEEE Conference on Computer Vision and Pattern Recognition CVPR 2012 (2012), long preprint arXiv:1202.2745v1 [cs.CV]
6. Ebrahimzadeh, R., Jampour, M.: Efficient handwritten digit recognition based on histogram of oriented gradients and svm. International Journal of Computer Applications 104(9), 10–13 (October 2014)
7. Fischer, A., Frinken, V., Bunke, H.: Hidden markov models for off-line cursive handwriting recognition. Handbook of Statistics: Machine Learning: Theory and Applications 31, 421 (2013)
8. Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H., Schmidhuber, J.: A novel connectionist system for unconstrained handwriting recognition. IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI) 31(5), 855–868 (2009)
9. Hinds, S.C., Fisher, J.L., D'Amato, D.P.: A document skew detection method using run-length encoding and the hough transform. In: 10th International Conference on Pattern Recognition (ICPR), 1990. vol. 1, pp. 464–468. IEEE (1990)
10. Ji, T., Kim, E., Srikantan, R., Tsai, A., Cordero, A., Wagner, D.: An analysis of write-in marks on optical scan ballots. In: Proceedings of the 2011 Conference on Electronic Voting Technology/Workshop on Trustworthy Elections. EVT/WOTE'11, USENIX Association, Berkeley, CA, USA (2011)
11. Keysers, D., Gollan, C., Ney, H.: Local context in non-linear deformation models for handwritten character recognition. In: 17th International Conference on Pattern Recognition (ICPR), 2004. vol. 4, pp. 511–514. IEEE (2004)
12. Kim, E., Carlini, N., Chang, A., Yiu, G., Wang, K., Wagner, D.: Improved support for machine-assisted ballot-level audits. In: Presented as part of the 2013 Electronic

Voting Technology Workshop/Workshop on Trustworthy Elections. USENIX, Berkeley, CA (2013), https://www.usenix.org/conference/evtwote13/workshop-program/presentation/Kim
13. Le, D.S., Thoma, G.R., Wechsler, H.: Automated page orientation and skew angle detection for binary document images. Pattern Recognition 27(10), 1325–1344 (1994)
14. Lecun, Y., Cortes, C.: The MNIST database of handwritten digits. http://yann.lecun.com/exdb/mnist/
15. Liu, C.L., Nakashima, K., Sako, H., Fujisawa, H.: Handwritten digit recognition: benchmarking of state-of-the-art techniques. Pattern Recognition 36(10), 2271–2285 (2003)
16. Niblack, W.: An introduction to digital image processing. Strandberg Publishing Company (1985)
17. Otsu, N.: A threshold selection method from gray-level histograms. Automatica 11(285-296), 23–27 (1975)
18. Plötz, T., Fink, G.A.: Markov Models for Offline Handwriting Recognition: A Survey. Int. Journal on Document Analysis and Recognition 12(4), 269–298 (2009)
19. Pratikakis, I., Gatos, B., Ntirogiannis, K.: Icdar 2013 document image binarization contest (dibco 2013). In: 12th International Conference on Document Analysis and Recognition (ICDAR), 2013. pp. 1471–1476. IEEE (2013)
20. Rabiner, L., Juang, B.H.: An introduction to hidden markov models. ASSP Magazine, IEEE 3(1), 4–16 (1986)
21. Sauvola, J., Pietikäinen, M.: Adaptive document image binarization. Pattern recognition 33(2), 225–236 (2000)
22. Serra, J.: Introduction to mathematical morphology. Comput. Vision Graph. Image Process. 35(3), 283–305 (Sep 1986)
23. Sezgin, M., et al.: Survey over image thresholding techniques and quantitative performance evaluation. Journal of Electronic imaging 13(1), 146–168 (2004)
24. Singh, C., Bhatia, N., Kaur, A.: Hough transform based fast skew detection and accurate skew correction methods. Pattern Recognition 41(12), 3528–3546 (2008)
25. Smith, E.H.B., Lopresti, D.P., Nagy, G.: Ballot mark detection. In: ICPR. pp. 1–4. IEEE (2008)
26. Smith, E.H.B., Lopresti, D.P., Nagy, G., Wu, Z.: Towards improved paper-based election technology. In: International Conference on Document Analysis and Recognition (ICDAR). pp. 1255–1259. IEEE (2011)
27. Smith, E.H.B., Nagy, G., Lopresti, D.P.: Mark detection from scanned ballots. In: Berkner, K., Likforman-Sulem, L. (eds.) DRR. SPIE Proceedings, vol. 7247, pp. 1–10. SPIE (2009)
28. Wang, K., Kim, E., Carlini, N., Motyashov, I., Nguyen, D., Wagner, D.: Operator-assisted tabulation of optical scan ballots. In: Presented as part of the 2012 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections. USENIX, Berkeley, CA (2012)
29. Xiu, P., Lopresti, D.P., Baird, H.S., Nagy, G., Smith, E.H.B.: Style-based ballot mark recognition. In: International Conference on Document Analysis and Recognition (ICDAR). pp. 216–220. IEEE (2009)