2009 10th International Conference on Document Analysis and Recognition

Scalable Feature Extraction from Noisy Documents

Loïc Lecerf and Boris Chidlovskii
Xerox Research Centre Europe
6, chemin de Maupertuis, F–38240 Meylan, France
chidlovskii,lecerf@xrce.xerox.com

Abstract

We cope with the metadata recognition in layout-oriented documents. We address the problem as a classification task and propose a method for automatic extraction of relevant features, in presence of content and structural noise, caused by scanning, OCR and segmentation problems. The method is based on the automatic analysis of documents and requires no particular preprocessing. The method mines the documents and determines frequent patterns, which are both literal patterns and their generalization. We also propose a sampling technique which processes a sample of documents and uses the Chernoff bounds to estimate the pattern frequency in the entire dataset.

As a number of frequent patterns as feature candidates grows, the method applies a scalable feature selection method to determine the most relevant features to a given classification task. A series of evaluations on two collections show that the method performs comparably to the manual work on rule writing made by domain experts.

1 Introduction

A large majority of techniques for document analysis assumes working not with documents directly, because of their intrinsic complexity, but with a reduced representative set of features. Such a transformation is called *features extraction*. If the features are carefully chosen, they are expected to extract the relevant information from the documents in order to perform a given task. One major problem is the number of features used. Data analysis with too many features generally requires a large amount of memory and the computation power. Feature extraction refers to constructing features that get around these problems while still describing the complex data with sufficient accuracy.

In this paper, we address the feature extraction from layout-oriented documents, in the context of metadata extraction and recognition task. Both rule-based and learning-based systems commonly use rules and regular expressions to analyze the text. As manually-crafted rules are very sensitive to *OCR errors*, string distance and equivalent dictionary-based techniques and fuzzy rules have been proposed [5, 6, 7, 8].

Feature extraction from noisy documents is even more challenging when the *content noise* (OCR errors) is accompanied with the *structural noise* (segmentation errors). In scanned and OCR-ed documents, document fragments may be under- or over-segmented. In electronic documents like PDF files or Web pages, the segmentation inconsistency takes place due to an ambiguous page layout, format conversion, etc.

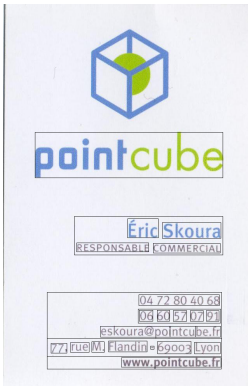
We propose a novel approach to the feature extraction from noisy documents. We mine the dataset for the frequent patterns and select those most relevant to the classification task. Our method is driven by few parameters, all being easily controlled by the user, like the list of separators and the minimum support threshold.

Motivation Example. Consider the business card image example in Figure 1.a. The OCR engine was used to extract raw textual data which was then labeled with metadata classes. Figure 1.b demonstrates the result, including OCR errors and segmentation inconsistencies in name, e-mail, street and URL fields.

Consider first the problem of recognizing the noisy e-mails. Correct e-mail sequences, due to their automatic recognition by mailers, DNS and network routers, should follow certain rules. The regular expression `'^.+@[^\.].*.[a-z]{2,}\$'` (in Python syntax) captures any valid, real life e-mail address. It permits any foreign character in the domain names and accepts anything before '@' symbol. The only roman alphabet restriction is in the domain field and the only dot restriction is that the dot cannot be placed directly after '@'.

The pattern is very general, however an error may make an e-mail sequence unparsable. Table 1 reports different examples from the Business Card collection (see Section 3 for details), where erroneous e-mail sequences are ubiquitous and vary from almost correct to fairly unreadable ones.

978-0-7695-3725-2/09 $25.00 © 2009 IEEE
DOI 10.1109/ICDAR.2009.227



a)

RAW DATA	LABEL
po!rlcube	
Eric Skouna	name
RESPONSABLE COMMERCIAL	role
0472804068	phone
06 60 57 07 91	mobile
eskoura@poi ntcu be.fr	e-mail
77, rue M. Aandin	street
-	
69003	zipcode
Lyon	city
www.pottubear	url

b)

Figure 1. Business card example. a) Card image, b) OCR-ed and extracted information.

Now consider three following regular expressions for finding email-like patterns in text:

1. Re_1 = `"\.\bcom\b"`: is 1 if the text contains the '.com' substring.
2. Re_2 = `"\@[a-z]{3,5}"`: is 1 if a text contains '@' followed by 3 to 5 characters in the **a** - **z** range.
3. Re_3 =`"\b\w\.\w\b"`: is 1 in the presence of two alphanumeric strings separated by a dot.

These three patterns have different generalization level, with Re_1 being the most specific and Re_3 being the most general. In cases when the correct e-mail pattern fails (lines (3) to (6) in Table 1), they give partial evidences for recognizing the strings as (erroneous) e-mail address.

Unlike e-mails, other typed data, like telephone/fax numbers, addresses etc., follow not one, but multiple different patterns and often assume human processing and understanding. When data heterogeneity is combined with data noise, the recognition task becomes even more difficult.

This however requires an appropriate abstraction of content and extracting features that give partial evidence for making a classification decision. Yet, feature extraction that tolerates a certain error/noise level is as difficult as writing advanced recognition rules. We therefore need a method capable to automatically mine the dataset for patterns that (1) tolerate a certain noise level and (2) serve as relevant features in training accurate learning models.

In this paper, we develop a method for automatic feature extraction from layout-oriented documents, in the presence of content and structural noise. The method is based on mining the documents and requires no special preprocessing. Its performance is comparable to the manual work

Content fragments	Parse	Re_1	Re_2	Re_3
`rigon@lan.fr`	yes	–	√	√
`krapfel@smith.umd.edu`	yes	–	√	√
`gawar@xgi .world.xerox.com`	no	√	√	√
`koura@poi ntcbe.fr`	no	–	√	√
`charles.huote isg roup.com`	no	√	–	√
`jim.cookvol com`	no	–	–	√

Table 1. Correct and noisy e-mails.

on rule writing accomplished by the domain expert. The method consists of two sequential steps as follows:

- *Frequent pattern extraction*: the input dataset is mined in the unsupervised mode, in order to extract frequent patterns, being both exact fragments and their generalization as regular expressions allowing to convey the possible noise.
- *Feature selection and fusion*: the number of frequent patterns may account for dozens of thousands or more patterns. For a given classification task, a large part of generated patterns is irrelevant; even a larger part of features is redundant. We use a scalable method for selecting an optimal feature set in the supervised mode. Optionally, we can further reduce the number of features by determining groups of complementary features and replace each group by one feature.

The first step is specific to layout-oriented documents; the way it works is based on the sequential nature of textual fragments and generalization of noise in text. The feature selection applies to any dataset, it selects a subset of relevant features for building robust learning models [1, 9]. The feature merge is optional, it targets the same goal as feature selection, it allows to speed up the learning process and improves the model accuracy.

2 Feature generation

The input documents are stored in the XML format; they reflect results (including errors) of the preprocessing steps, such as format conversion, segmentation, OCR, logical analysis, etc. For each XML node, we extract the content associated with it, and additionally the class label if available. Additionally, for any given node we extract its neighbors in the XML structure. The neighbor's content will help us generate complex features conditioned by the neighborhood relationship.

We mine the noisy documents for frequent patterns and address the following issues:

Frequent and contiguous patterns. First, we extract patterns which are literal fragments occurring k times (*min-*

imum support) in the dataset. Unlike the conventional pattern mining [2, 4], the patterns we extract are contiguous.

Generalization and composition. We generalize literal patterns in order to convey the content noise. The generalization takes place on different levels and captures a larger class of sequences. More complex patterns are obtained by composition from simpler ones.

Efficiency. All potential patterns form a complex lattice [4] whose size grows exponentially in the number of basic elements. Naive approaches to finding the frequent patterns on the lattice might be prohibitively expensive. First, we propose an efficient method which excludes multiple scans of the dataset when composing new patterns. The method works well when the dataset fits in the main memory where the evaluation can be quickly done. In the case of very large datasets, we propose a *sampling technique*. We sample documents from the dataset and apply the feature extraction on the sample and use the *Chernoff bounds* [3] to estimate the pattern frequency in the entire dataset.

2.1 Pattern extraction algorithm

The algorithm for extracting frequent patterns from a set D of noisy documents works in the recursive manner and copes with the following sets of patterns:

1. *Separator set.* Separators split textual sequences into tokens. By default, this set includes standard separators like '\n', '\t', ',', etc. The set can be extended to accommodate specific syntactic patterns in the dataset.

2. *Factual patterns.* Using the full dataset of textual sequences, the generator extracts the literal elements occurred at least k times: 1) the set of all alphanumeric tokens, like 'Fax', 'Lyon', 'E-mail', etc. (pattern `'\b\w+\b'` in Python); 2) the set of non alphanumeric character sequences between two separators.

3. *Generalized patterns.* According to the generalization principle, an upper-case, lower-case character and a digit are generalized as `'[a-z]'`, `'[A-Z]'` and `'[0-9]'`, respectively. Thus, words 'fr' and 'Phone' are generalized as `'[a-z]{2}'` and `'[A-Z][a-z]{4}'`. The set of literal and generalized patterns form the basic pattern set P.

   The method allows for different levels of generalization from literal patterns, including the alphabet enlargement, the occurrence extension or both. For textual pattern 'Phone', the generalized patterns are any of the following: `'[A-Z][a-z]{4}'`, `'[A-Za-z]{5}'`, `'\w{3,5}'`, `'\w+'`. No basic nor generalized patterns can match the empty string. A pattern may have multiple matches in the dataset; each match is given by the pair (b, e) indicating the start and end positions of the match.

4. *Compound patterns.* A compound pattern is a concatenation of two or more basic patterns. We retrieve all compound patterns occurred at least k times in the documents. Using the match sets M for all basic elements, we apply a recursive algorithm which determines all frequent compound patterns.

The frequent pattern generator calls recursively Algorithm 1 on each basic pattern $p \in P$. It profits from the regular pattern match on arbitrary strings, according to which $match(p_1, d)$ =(a, b) and $match(p_2, d) = (b + 1, c)$ implies $match(concat(p_1, p_2), d) = (a, c)$, where d is an input document. The algorithm tries to compound a new pattern by checking whether the ending position of a current pattern is followed by the beginning position of a basic one. The algorithm is the following:

Algorithm 1 Find all frequent patterns starting with a given pattern

Require: cur-current pattern, $M_c = \{(b_i, e_i)\}$ - the match set of cur, P - set of basic patterns
Ensure: RES - all frequent patterns prefixed with cur

```
1:  RES := ∅
2:  for all p ∈ P do
3:     Let M_p = {(b_j, e_j)} be the match set of p
4:     M := {(b_i, e_j) | e_i + 1 = b_j, e_i ∈ M_c, b_j ∈ M_p}
5:     if |M| > k then
6:        new := concat(cur, p)
7:        RES := RES ∪ new ∪ Algorithm1(new, M)
8:     end if
9:  end for
10: return RES
```

The set P of basic patterns is extracted by a unique scan of the dataset. Instead, the method avoids additional scans of the dataset with compound patterns. For each new compound pattern new, its match set M is obtained from match sets M_c and M_p of its components. Moreover, its size $|M|$ is lower that any of these two, $|M| \leq min(|M_c|, |M_p|)$.

Figure 2 plots two measures of the feature generation for two collections used in the evaluation, Business Card and CPO (see Section 3 for detail). Plots report the processing time and the number of frequent patterns, for the varying number of documents.

The feature generation method prunes off the infrequent patterns and excludes any extra scans of the dataset with compound patterns. However, the match in line (5) of Algorithm 1 might be prohibitive if 1) the dataset is too large, or 2) the size of match sets M_c and M_p is comparable to the dataset size.

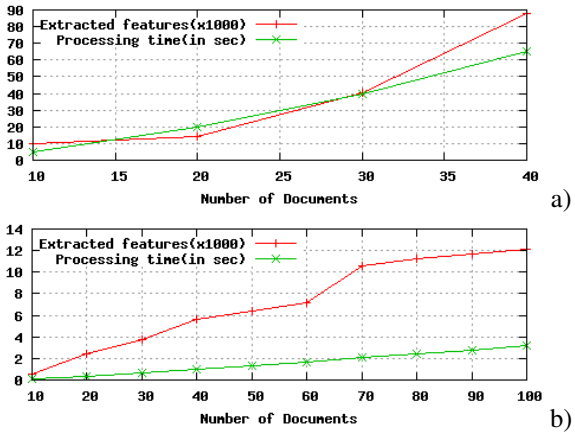


Figure 2. Feature generation processing. a) BizCard collection, b) CPO collection.

2.2 Sampling techniques

If the document set is too large, we propose a sampling technique to obtain a quick estimation of the frequent pattern set, where additional checks can be performed to finalize the pattern set. In order to obtain an estimation of the frequent patterns without examining the entire dataset, we use the *additive Chernoff bound*; it tests a hypothesis based on the sum of observations to estimate the range of the pattern match from a sample with a high statistical confidence.

Let X be a random variable whose spread is R. Suppose that we have n independent observations of X, and the mean is μ_n. The Chernoff bound states that with probability $1-\delta$, the true mean μ of X is within $\mu_n - \epsilon \leq \mu \leq \mu_n + \epsilon$, where $\epsilon = R\sqrt{\frac{ln(1/\delta)}{2n}}$.

The Chernoff bound estimation is applied to the frequent pattern sampling as follows. Given a set D_S of n samples, $n \ll |D|$ and the minimum support k, a pattern p is *frequent* with probability $1-\epsilon$ if $k \leq \mu_n - \epsilon$ and is *infrequent* with probability $1-\epsilon$ if $\mu_n < k - \epsilon$, where μ_n is the pattern match in the sample ste D_S. *Ambiguous* are patterns whose matches in the sample satisfy $k - \epsilon < \mu_n < k + \epsilon$; they remain undecided and might need a further examination.

The Chernoff bound is independent of the probability distribution that generates the observations X, as far as such probability distribution remains static. The number of ambiguous patterns highly depends on the value of ϵ which itself is a function of the sample size n. In order to further reduce ϵ, we derive the most restricted spread R for the match of each compound pattern. Following the Apriori (monotonicity) property [2], the match of a compound pattern is always less than or equal to the minimum match of each basic pattern in it.

The sampling method for frequent patterns is as follows:

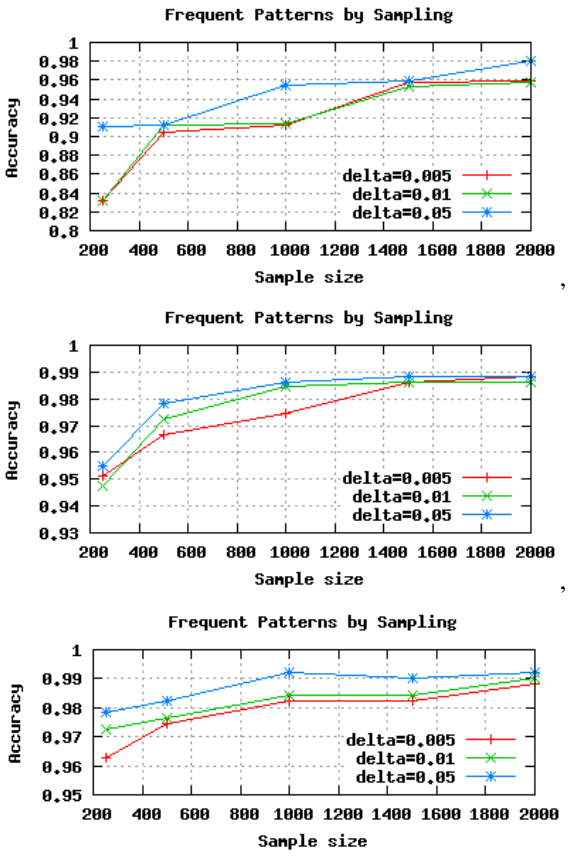


Figure 3. Accuracy of frequent feature detection by sampling, a) k=2, c) k=10, c) k=50.

1. While scanning the entire dataset D, take a random sample D_S and find matches $M_D(p)$ and $M_{D_S}(p)$ for each basic pattern $p \in P$ in D and D_S.

2. Identify the frequent pattern set F_S on the sample set D_S using Algorithm 1 and basic pattern matches in D_S, with minimum support $k' = k\frac{|D_S|}{|D|}$.

3. For each compound pattern $cp \in F_S$, estimate its expected value $\mu = \frac{|D|}{|D_S|}\mu_{D_S}$ and the Chernoff bound ϵ on the entire dataset D with $R = min_{p \in cp} M_D(p)$.

4. Select frequent patterns with $\mu - \epsilon > k$. Optionally, verify the ambiguous patterns on the entire dataset.

Figure 3 reports the evaluation of the frequent pattern sampling on the Business Card collection. In each test, we measure the percentage of correct decisions on patterns, given by $A = \frac{q+r}{T}$, where q and r are the number of correctly detected frequent and infrequent patterns, T is the total number of patterns. Quantity $1 - A$ refers to the percentage of misclassified and ambiguous patterns. The figure

reports the accuracy A for different values of statistical tolerance δ and threshold k. The dataset includes about 4000 elements (observations), the sampling is run with values n between 250 and 2000.

2.3 Feature selection

To select among a large number of frequent patterns, we remove features which are irrelevant and redundant for the classification task. We first measure the correlations between feature candidates. Then we use the multi-class extension [1] of the Fast Correlation-Based Filtering [9] to select optimal subsets for robust learning models.

An important extension for feature generation is the use of element neighborhood. Conventionally, features link characteristics of the current observation to its label. This approach can be enlarged to neighbor observations. This helps determine a correlation between the label and characteristics of observations preceding or succeeding the current one. A typical example is to link the rule for phone and fax numbers with the presence of terms 'Phone' and 'Fax' in the observation preceding the current one.

3 Evaluation

This sections completes the tests presented in Section 2. Two different corpora have been used in the evaluation. **BizCard** is a collection of 100 Business Card images, scanned, OCR-ed and manually annotated with metadata labels (see example in Figure 1). **CPO** is a collection of 40 PDF documents with metadata annotation, created in the framework of VIKEF EU project (see `http://www.vikef.net`).

All tests are run in the cross validation mode. The CPO dataset has been split in 20 folds, 40 folds are used in for BizCard dataset. For each folding, the feature extraction , applied to the training set, produces a feature set which is used to train a logistic regression model. The average over all folds is reported as the model accuracy for the collection.

Table 2 reports some results of the automatic feature extraction and compares them to the feature sets manually crafted by experts (133 features for Bizcard and 42 features for CPO collection). Training logistic regression classifiers with these sets result in accuracy 70.97% and 87.68%, respectively. The automatic feature generation is run with $k = 2$; it produces 12,107 frequent patterns for BizCard and 87,833 for CPO. The neighborhood width in tests was $N = 0$, 1 or 2. Feature selection with $N = 2$ makes an average of 311,1 features for BizCard and 423,2 for CPO (with the length of compound patterns limited to 40). Additional tests keep the top 200 features only. As the table shows, the automatic feature extraction tuned with few parameters allow achieve the classification accuracy comparable to the manually crafted features.

Collection	Manual feat	Automatic feature extract		
	Accuracy	k	N	Accuracy
BizCard	**70.97**	2	0	61.26
BizCard	**70.97**	2	1	69.37
BizCard	70.97	2	1, top 200	**71.02**
CPO	**87.68**	2	0	60.73
CPO	**87.68**	2	1	84.93
CPO	87.68	2	2	**87.77**

Table 2. Evaluation results.

4 Conclusion

We proposed a method for the automatic feature extraction from documents with content ans structural noise. The method combines the frequent pattern extraction from documents with the scalable feature selection. Evaluations show that the method performs comparably to the manual work on patterns and rule writing.

Evaluations also show that for small values of minimal support k, the current method may suffer from generating too many frequent patterns. In the future, we plan to improve the method for the supervised mode, where frequent pattern extraction and feature selection get executed not sequentially but jointly. Using the feature selection criteria when mining documents for frequent patterns can reduce considerably the feature set and enhance the scalability.

Acknowledgment. This work is partially supported by the ATASH Project co-funded by the French Association on Research and Technology (ANRT).

References

[1] Boris Chidlovskii and Loïc Lecerf. Scalable feature selection for multi-class problems. In *Proc. ECML PKDD*, pages 227–240, 2008.

[2] Bart Goethals. Survey on frequent pattern mining. HIIT Basic Research Unit. University of Helsinki, 2003.

[3] Torben Hagerup and C. Rüb. A guided tour of chernoff bounds. *Inf. Process. Lett.*, 33(6):305–308, 1990.

[4] Jiawei Han, Hong Cheng, Dong Xin, and Xifeng Yan. Frequent pattern mining: current status and future directions. *Data Min. Knowl. Discov.*, 15(1):55–86, 2007.

[5] Kwangbaek Kim, Jaehyun Cho, and Amsuk Oh. Recognition of english business cards using enhanced hybrid network. In *Advances in Neural Networks, LNCS*, volume 3497, pages 110–121, 2005.

[6] Colour Business Card Scanner. http://www.cardscan.com.

[7] Xiang Tong and David A. Evans. A statistical approach to automatic ocr error correction in context. In *Proc. 4th Workshop on Very Large Corpora*, pages 88–100, 1996.

[8] Qivind D. Trier, Anil K. Jain, and Torfinn Taxt. Feature extraction methods for character recognition - a survey. *Pattern Recognition*, 29(4):641–662, 1996.

[9] Lei Yu and Huan Liu. Efficient feature selection via analysis of relevance and redundancy. *J. Mach. Learn. Res.*, 5:1205–1224, 2004.