A System for Address Extraction from Facsimile Images

Craig R. Nohl **Jan I. Ben** **Dar-Shyang Lee**[1] **Christopher J. C. Burges**

Lucent Technologies Bell Labs
101 Crawfords Corner Road
Holmdel, NJ 07733 USA
nohl@lucent.com
ben@lucent.com
burges@lucent.com

Abstract

We describe a prototype system capable of extracting machine print addresses from fax images of English language business letters and fax cover sheets. The system automatically orients incoming page images, locates and parses machine printed addresses, and classifies each address as one of {sender, recipient, other}. We present results of preliminary performance tests, and discuss potential improvements.

1 Introduction

Document image understanding systems are useful to the extent that they can automatically extract *actionable* information from document images. In recent years, systems have been designed for progressively broader applications, including mailpiece routing, amount reading from financial documents, data entry from forms, invoice processing, and routing of business letters [1–7]. Lii and Srihari [8] consider the extraction of name and address information from fax cover sheets where field identifiers are present.

Practical systems were first developed for OCR-ready and other tightly structured documents, where the geometric and logical layout do not vary appreciably among documents of a given class. For these documents, the locations and semantics of document fields may be treated as known in advance.

More recently attention has been directed toward extraction of specific information from less structured documents, where different documents belonging to the same class may adhere only approximately to rules for geometrical and logical layout.

U.S. personal bank checks are a good example of a relatively tightly structured document class. While personal checks are not OCR-ready, and differ widely in appearance, they do conform to size constraints, and the locations of specific fields are specified with respect to a reference point on the physical document.

A *semi-structured* document class has members that exhibit statistical regularities in contents and layout, rather than uniformity. U.S. *commercial* bank checks are a good example. With the exception of the MICR (magnetic ink) line at the bottom, the layout of the document is up to the printer. Field locations do not conform to well-defined rules. However, some common practices are observed in the layout. Signatures are usually on the right-hand side of the check, toward the bottom. The date is most often in the upper half of the check, and frequently in the right half. The courtesy amount usually appears above the signature, and tends to be on the right hand side. The payee name (and sometimes the payee address) is usually in one of several general locations.

In this paper, we consider the problem of locating and determining the roles of addresses in images of two other semi-structured document classes: business letters and fax cover sheets.

Addresses provide important information about the documents containing them: they identify persons or organizations that have a variety of roles with respect to a document: author, transmitter, recipient, contact, etc. Addresses can thus be important for routing, indexing, and retrieving documents in a spectrum of real-world applications. This is true in particular of documents typically transmitted by facsimile.

Business letters commonly contain the same set of components: organization name and logo, sender address, recipient address, date, salutation, body, closing ("Sincerely"), and signature block containing a signature plus machine printed text identifying the signer. Beyond this there are no hard rules for where the components appear or how they are formatted. Sender address information may appear at the top of bottom of a page; on the left or right side; in block, multi-column, or single line format. Sender

[1]Current address: Ricoh Silicon Valley, Inc., 2882 Sand Hill Road, Suite 115, Menlo Park, CA 94025 dsl@rsv.ricoh.com

address information may appear in more than one place. Recipient addresses may also appear in several locations, including locations typical of sender addresses. Business letters also frequently contain other address information that is associated with neither sender nor recipient.

Fax cover sheets exhibit an even wider variety of contents and layouts. Some contain no more than a handwritten name and fax number. Others look like letterhead from the sender's organization, perhaps with fields added to contain recipient address information. Still others consist of a stick-on form (with fields for sender and recipient information) applied to another document page. Many contain a mixture of handwritten and machine printed information; others are machine-generated on desktop computers, and are entirely machine printed. Some are cover sheets recycled from a fax transmission in the opposite direction ("FROM" overwritten with "TO" and vice versa). Sender and recipient address information may be formatted in distinct blocks or not. They may occupy side-by-side columns, or instead fill groups of adjacent rows in a two-column tabular format.

We describe a system designed to extract and classify machine print addresses from fax images. Our approach, which uses both lexical and geometric layout cues to identify addresses, promises good results even when image quality limits OCR accuracy.

We have focused initially on extracting U.S. addresses for senders and recipients from English language business letters and fax cover sheets. More specifically, we attempted to extract all addresses that contain at least a P.O. Box or street address plus city and state, and to classify all such addresses as one of {`SENDER, RECIPIENT, OTHER`}. For fax cover sheets we attempt more generally to extract any address information associated with a "To:" or "From:" identifier (or equivalent).

Address extraction is potentially useful in a number of applications. In a multi-media messaging system, where users' mailboxes may contain voice messages, email, and fax messages, address extraction supports a more friendly user interface for those who have to deal with large numbers of messages. For a desktop mailbox interface, the sender name can be included in message summaries, and messages can be listed in order priority according to sender. When accessing messages via voice interface (e.g., from a cell phone), such features are even more important to the user's efficiency. It is also possible to alert on the receipt of messages from particular senders, or to forward copies to others who are able to respond more quickly.

In the following sections, we give an overview of our system, discuss keyword spotting and the structure of address grammars, and report results of preliminary experiments.

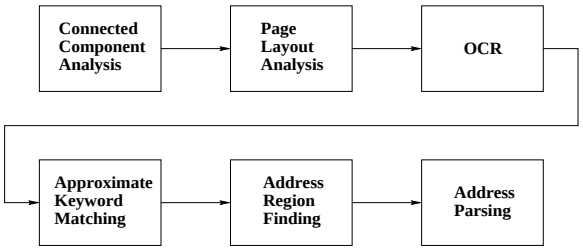

Figure 1: System Processing Overview

2 System Overview

Figure 1 shows the overall processing flow of our system.

Processing on standard or fine resolution fax images begins with connected component analysis, followed by a page layout analysis using algorithms that do not require prior knowledge of text line orientation [9]. For multi-lingual image streams, script identification and page orientation detection are performed using the methods described in [10]. OCR on lines of machine printed text is performed using the system described in [11].

OCR output is then examined in the context of the page layout to localize and parse potential addresses. Addresses are then classified according to their type.

3 Address Extraction

Addresses are extracted using a three step procedure:

- OCR output is processed to spot keywords commonly appearing in addresses, such as street suffixes ("Street", "Avenue", "Circle") and state names, and patterns typical of ZIP Codes and telephone numbers.

- Using the keywords found, one or more regions are defined such that each is very likely to contain all the words of an address.

- Words within each region are parsed according to an address grammar. Parsing assigns an address "part of speech" (e.g., `CITY_WORD`, or `JUNK` for non address words) to each word in the region, thereby locating one or more addresses. Parsing returns a score reflecting the degree of confidence in the identification.

The subsections below contain a more detailed description.

3.1 Approximate Keyword Spotting

We use keyword spotting to limit computation by limiting the portions of a document page subjected to parsing.

Since fax images are usually of relatively low resolution (200x100 dpi or 200x200 dpi) and may be imaged from poor quality hard copies, it is important that keyword spotting be tolerant of OCR errors; therefore, we used the fast approximate matching algorithm [12]. Our implementation permits matching on any number of patterns with integer insertion, deletion, and substitution costs and maximum edit cost specified separately for each pattern. It is also possible to specify that certain characters in a pattern be matched exactly. The maximum edit distances are tuned to optimize word-spotting performance for the combination of OCR engine and typical image quality.

Patterns are associated with lexical tokens, such that one or more tokens are associated with a word of OCR output through matches.

Figure 2 shows a portion of the keyword matching specification used for U.S. address extraction.

Table 1: Tag types for keyword matching

Tag	*Matches*
TELNTAG	Telephone number
ZIPTAG	ZIP Code
DIRTAG	Directional ("North")
SUFXTAG	Street suffix ("Avenue")
UNITTAG	Building part identifier ("Apt")
POBOXTAG	Post Office Box identifier
TELTAG	Telephone identifier ("Tel","Fax")
CITYTAG	City name
STATETAG	State name or abbreviation
EMAILTAG	Email identifier ("email")
EMAILADTAG	Email address
FROMTAG	Sender identifier ("From")
TOTAG	Recipient identifier ("To")
COMPANYTAG	Firm identifier ("Company")
GCAPWD	General capitalized word
GDIGSTR	General digit string

Table 1 shows a listing of tag types used for keyword matching. Note that in addition to tags intended to match the common components of addresses, there are also tags for *identifiers* – words that communicate the purpose of adjacent information in a pre-printed address ("Fax", "email"), or the purpose of fields on a fax cover sheet ("To", "From", "Company"). Finally, there are two general-purpose tags for capitalized words and strings of digits. These help in the subsequent parsing of address

```
Tag TELNTAG 1.0
 "(\d\d\d) \d\d\d[ \p]\d\d\d\d" \
  1  2  2  2  0   \
 "\d\d\d[ \p]\d\d\d[ \p]\d\d\d\d" \
  1  2  2  2  0

Tag  ZIPTAG  1.0
 "\d\d\d\d\d\p\d\d\d\d" 1  2  2  2  0   \
 "[\dO1][\dO1][\dO1][\dO1][\dO1][ \p]" \
  1  2  2  0  0

Tag SUFXTAG 1.0
 "Street"   1  1  1  1  0   \
 "S[tT]"   1  1  1  0  0  \
 "Road"    1  1  1  0  0   \
 "R[dD]"   1  1  1  0  0  \
 "Avenue"   1  1  1  1  0   \
 "Ave"   1  1  1  0  0  \
 "Drive"   1  1  1  0  0  \
 "D[rR]"   1  1  1  0  0

Tag UNITTAG 1.0
 "APT"   1  1  1  0  0  \
 "SUITE"   1  1  1  1  0  \
 "ROOM"   1  1  1  0  0

Tag TELTAG  1.0
 "Fax"  1  1  1  0  0   \
 "Tel"  1  1  1  0  0   \
 "Telephone"  1  1  1  2  0  \
```

Figure 2: Part of the keyword matching specification for U.S. addresses. Fields within each pattern specification are: pattern, substitution cost, insertion cost, deletion cost, maximum edit distance, ignore case flag. The sequence \d matches any digit; the sequence \p matches punctuation.

forms.

3.2 Address Region Finding

The purpose of address region finding is to reduce overall computation by applying full-fledged address parsing only to those regions likely to contain an address.

In our system, address region finding and address extraction are currently limited to the domain of a single text block, as found by the page layout analysis algorithm.

In an early implementation, an address region was deduced from the locations and lexical tags for keyword matches within a text block, based on the structure of the grammar ultimately used for parsing the address. The grammar was analyzed to determine for each type of lexical tag the maximum number of address words that could precede and follow a word with that tag. Thus, each keyword match defined an address region ("window"). The union of such windows (for all keyword matches in the block) was the address region.

Subsequently, we have implemented grammar-based address region finding. This approach has two advantages: first, it can potentially avoid generating address regions where only isolated keyword matches are present (e.g., isolated city names in the text of a letter); second, it is capable of using geometrical layout cues to find candidate regions even when there are no keyword matches.

In the grammar-based approach, the results of page layout analysis and text interpretation plus keyword spotting are used to generate a stream of lexical tokens that are parsed by the grammar.

- *Geometrical layout* tokens include delimiters for text blocks and text lines, plus markers to indicate the alignment of the current line with the previous line. Geometrical layout tokens are shown in Table 2.
- *Word* tokens are either the specific lexical tokens generated by keyword matching, or a generic ("other") word token.

Table 2: Geometrical layout tokens

Token	*Meaning*
EB	block delimiter
EL	line delimiter
LA	line left-aligned with preceding line
GL	matches any geometrical layout token

The grammar is a stochastic regular grammar for the occurrence of one or more addresses in a text block, formatted either in common block formats (as in address blocks in business letters) or in *in-line* format (*"... next week. Please send any additional information to Craig Nohl, Lucent Technologies Bell Labs, PO Box 3030, Holmdel NJ 07733"*).

A grammar is defined as a 4-tuple $G = \{\mathcal{N}, \mathcal{T}, \mathcal{P}, \mathcal{S}\}$, where $\mathcal{N}$ is a set of *nonterminal* symbols, $\mathcal{T}$ is a set of *terminal symbols*, $\mathcal{P}$ is a set of *production rules*, and $\mathcal{S} \in \mathcal{N}$ is a special nonterminal called the *start* symbol. Conventionally, elements of $\mathcal{N}$ are written as upper case Roman letters (A,B,C,...) and elements of $\mathcal{T}$ are written as lower case Roman letters (a,b,c,...). *Strings* of symbols from $\mathcal{V} = \mathcal{N} \cup \mathcal{T}$ are written as lower case Greek letters ($\alpha, \beta, \gamma, ...$). Production rules are of the form $\alpha \rightarrow \beta$.

The set of all *strings* of finite length over an alphabet $\mathcal{V}$ is denoted $\mathcal{V}^*$. $\mathcal{V}^*$ includes the null string λ. When $\mathcal{P}$ contains the production rule $\alpha \rightarrow \beta$, the concatenation of strings $\gamma_1\alpha\gamma_2$ *derives* the string $\gamma_1\beta\gamma_2$ in the grammar G, written $\gamma_1\alpha\gamma_2 \Rightarrow \gamma_1\beta\gamma_2$. When $\sigma_1 \Rightarrow \sigma_2 \Rightarrow \cdots \Rightarrow \sigma_n$, the string σ_1 is said to *ultimately derive* the string σ_2. The set of all *sentences*, or finite-length strings of terminals ultimately derived from $\mathcal{S}$ in G, is called the *language* accepted by G, $L(G)$.

Each derivation of a sentence from the start symbol is a *parse* of the sentence.

A *regular* grammar can be expressed such that all production rules are of the form $A \rightarrow aB$ or $A \rightarrow a$.

A *stochastic* grammar assigns a probability to each production rule, and thus induces a probability measure on derivations, parses, and sentences. In this paper, we speak instead of the *cost* of a derivation or parse. Costs are to be thought of as the negative logarithms of probabilities, $C = -\log P$; however, the use of this language is intended to suggest that we may not have an estimate of a true probability. The cost of a parse is the sum of the costs of its individual derivations; a parse with a smaller cost (higher probability) is *better* than a parse with a larger cost.

3.3 Structure of the Address Grammar

The address grammar is used to determine the best (lowest cost) parse of the lexical token stream. This parse results in the assignment of each text word to an address component type (i.e., an address "part of speech"), or to the non- address component type `JUNK`. When the parse is complete, special nonterminals `BEGIN` and `END` can be associated with positions in the lexical token stream. The intervening words constitute a candidate address region. If the cost of the best parse is below a threshold, the address region is accepted for further analysis.

We found it convenient to specify the grammar

hierarchically, as the composition of three components:

- *Full Address to Line Grammar.* The structure of a full address, expressed as allowed sequences of address line types.
- *Line to Address Component Grammar.* The structure of each line type, expressed as allowed sequences of address components.
- *Address Component to Word Grammar.* The structure of each address component, expressed as allowed sequences of Word lexical tokens.

This corresponds to partitioning of the grammar's production rules into the three sets above, and mapping its nonterminals into two sets $\mathcal{L}$ (corresponding to address line types) and $\mathcal{C}$ (corresponding to address component types). The Full Address to Line Grammar has production rules $\mathcal{S} \rightarrow \alpha$, where $\alpha \in \mathcal{L}^*$. The Line to Address Component Grammar has production rules $\alpha \rightarrow \beta$, where $\alpha \in \mathcal{L}^*$ and $\beta \in \mathcal{C}^*$. Finally, the Address Component to Word Grammar has production rules $\beta \rightarrow \gamma$, where $\beta \in \mathcal{C}^*$ and $\gamma \in \mathcal{T}^*$ is a string over the terminals.

It is convenient to summarize our grammar in an equivalent, but more compact, notation. We use an extended version of the familiar notation for regular expressions. The expression $A : \mathcal{R}$, where A is a single nonterminal and $\mathcal{R}$ is a regular expression constructed according to Table 3, is equivalent to the set of production rules $\{A \rightarrow \alpha \mid \alpha \; matches \; \mathcal{R}\}$. Note that terms in the regular expression $\mathcal{R}$ contain cost attributes; the cost for a production rule is the sum of the costs associated with the matching terms in $\mathcal{R}$.

Table 3: Notation for regular expression operations

Symbol	*Operation*
//	delimit regular expression
{}	enclose multi-character token
|	OR
?	zero or one occurrences of previous expression
*	zero or more occurrences of previous expression
()	grouping of operations
<>	cost for preceeding expression

A simple example of a Full Address to Line grammar is shown in Figure 3. Note that four types of tokens are accepted at this level of the grammar: geometrical layout tokens, line type tokens (summarized in Table 4), the `JUNK` token (intended to match non-address text), and the auxiliary tokens `BEGIN` and `END`, which delimit the address region.

```
{START}:
  / (({JUNK}<.05>)|{GL})*

  {BEGIN}
  ({EL}|{LA})?
  (({BOXLN}<3>)?
   ({STLN}<.1>({EL}{LA})? ({SDLN}<1.0>)?)
   )
  {EL}? {LA}?
  {CSZLN}
  {EL}? {LA}?
  ({TELN}<0.7> {EL}? {LA}?)*
  {END}

  (({JUNK}<.1>)|{GL})* /
```

Figure 3: A simple Full Address to Line grammar. Additive costs associated with tokens or compound expressions are shown in angle brackets (<>).

Table 4: Address line types

Symbol	*Line Type*
`TOLN`	Cover sheet TO line
`FROMLN`	Cover sheet FROM line
`COMPLN`	Company name
`BOXLN`	PO Box line
`STLN`	Street line
`SDLN`	Street secondary line
`CSZLN`	City/state/ZIP line
`TELN`	Telephone/fax line
`EMAILN`	Email line

A sample Line to Address Component grammar is presented in Figure 4.

The main purpose of the Address Component to Word grammar is to associate keyword matching tags and unmatched words with address components, which generally may consist of one or more words. A sample Address Component to Word grammar is shown in Figure 5. Costs in this grammar correspond to the probabilities that a particular address component will contain a given number of words, or that keyword match will actually occur where appropriate. One new token is present in this grammar: the null token `Eps` representing null input or output. This occurs because no input token is required to generate the `BEGIN` and `END` tokens that are accepted by the top level grammar to delimit an address region.

```
{CSZLN}:/ ({CITY}{STATE}{ZIP}) /
{BOXLN}:/ ({POBOX}{BOXNUM}) /
{STLN}:/ ({STNUM}{STNAME}{SUFX}<0.4>) |
         ({STNUM}{PRFX}{STNAME}{SUFX}<2.2>) |
         ({STNUM}{STNAME}{SUFX}{UNITNUM}<2.2>) |
         ({STNUM}{STNAME}{SUFX}{PRFX}<2.2>) /

{TELN}:/ ({TEL}{TELNUM}<0.9>) |
         ({TELNUM}<.5>) /

{SDLN}:/ ({UNIT}{UNITNUM}) /
{EMAILN}:/ ({EMAIL}{EMAILAD}) /
{JUNK}:/ {JUNK} /
{GL}:/ {GL} /
{BEGIN}:/ {BEGIN} /
{END}:/ {END} /
```

Figure 4: A simple Line to Address Component grammar, expressing line types in terms of sequences of address components.

```
{STNUM}:/ ({GDIGSTR} |{w}<1> ) /
{STNAME}:/  ({GCAPWD} |
 ({GCAPWD}{GCAPWD})<1.1> | {w}<1.5> |
 ({w}{w})<2> | ({w}{w}{w})<4> ) /
{SUFX}:/ ({SUFXTAG} | {w}<1>) /
{PRFX}:/ ({PRFXTAG} | {w}<1>) /
{CITY}:/ ({CITYTAG} | {GCAPWD}<1> |
 ({GCAPWD}{GCAPWD})<1.5> | {w}<2> |
 ({w}{w})<3>) /
{STATE}:/ ({STATETAG} | {GCAPWD}<3> |
 {w}<4> | ({w}{w})<5>) /
{ZIP}:/ {ZIPTAG} | {GDIGSTR}<1> |
 {w}<2.5> /
{TEL}:/ {TELTAG} | {w}<5> /
{TELNUM}:/ ({TELNUMTAG} |
 ({GDIGSTR}{GDIGSTR})<5> |
 {w}<8> | ({w}{w})<8>) /
{GL}:/ {GL} /
{BEGIN}:/ {Eps} /
{END}:/ {Eps} /
```

Figure 5: Part of a simple Address Component to Word grammar, expressing forms of allowed address components in terms of keyword matches and unmatched words.

3.4 Grammar Training

To date, most of the cost parameters in our grammar have been hand-tuned. However, it is possible in principle to train them from ground truthed data by standard techniques, as follows.

For the training data set, ground truthing assigns a part of speech for each word, and yields a token stream consisting of these plus geometric layout tokens capturing text block structure, line breaks, and alignment of succesive text lines. In addition, the `BEGIN` and `END` tokens are inserted to delimit the actual address. The token stream for all truthed address regions are parsed by the top two layers of the grammar (Full Address to Line and Line to Address Component), and the uses of each production rule are counted. For each set of production rules sharing the same nonterminal on the left-hand side, the counts are used to estimate the probability of each possible production given the nonterminal. The negative logarithms of these counts are the costs of the productions.

This leaves the problem of determining the set of production rules. We did this by trial and error on a training database. We have not considered techniques for automated grammar inference.

3.5 Address Parsing

Once address regions have been found, they are parsed more carefully with substantially the same grammar used for address region location, but with one important change to the parsing procedure. In address region location, the costs for associating a particular interpreted text word with an address component type depended only on the set of matching keywords. No effort was made to assess the *degree* of matching. We attempt to remedy this deficiency by modeling the probability of correct matches for each address component type.

For each of the address component types {`POBOX, DIR, SUFX, UNIT, CITY, STATE, TEL`}, word interpretations are compared with a list words that may appear in an address component of that type. A proximity $d \in [0,1]$ by a fast approximate string matching algorithm, with matching cost equal to $-\log(d+\epsilon)$. Proximity equals 1 for a perfect match, and the small positive number ϵ corresponds to the probability of a match when the proximity is zero.

Address components that are normally numbers, e.g., `STNUM, UNITNUM, BOXNUM, ZIP, TELNUM`, receive a matching cost $C = -\log P_{char\ class} - \log P_{string\ length}$. Here $P_{char\ class}$ models the probability of the observed sequence of character classes {ALPHA,DIGIT,OTHER} in the OCR output given that the actual word is a string of digits; alphabetical characters and punctuation easily confused with digits are treated as special cases. $P_{string\ length}$ models

x coord of block ULH corner (x_{ULH})
y coord of block ULH corner (y_{ULH})
$N_{blocks\ to\ left} - N_{blocks\ to\ right}$ using x0
$N_{blocks\ above} - N_{blocks\ below}$ using y0
Text block area
Text block aspect ratio
$(y_{LRH} - y_{ULH})/(x_{LRH} - x_{ULH})$
Number of words in text block
Number of lines in text block
Number of left-aligned lines in text block
Fraction of block's words in address region
Number of TEL tags
Number of EMAIL tags
Number of TO tags
Number of FROM tags

Figure 6: Features used in address type classification.

the prior probability of occurrence of a digit string of specified length for an address component of the specified type.

Street names (STNAME) are currently modeled as capitalized alphabetical strings, using an approach similar to that described above for numbers.

The parsing is of a new token stream, consisting of a sequence of word interpretations and geometric layout (GL) symbols in reading order, as produced by the OCR subsystem. The Address Component to Word grammar of Figure 5 is in effect replaced by a version without the lexical tags from keyword matching: thus the "production rules" of Figure 5 contain only generic word tokens ({w}) on their right hand sides. As generic word tokens are matched with word interpretations from the input token stream, matching scores are computed dynamically using the algorithms described above.

The overall parsing algorithm computes the lowest-cost parse for the full grammar. As before, in the course of parsing an address component type (or JUNK) is assigned to each word in the address region. The identification of an address is accepted or rejected according to the cost of the parse.

4 Address Type Classification

Addresses found by the procedures described above are classified as SENDER, RECIPIENT, or OTHER addresses. Our approach to classification uses only the geometrical layout features of address regions, plus limited lexical information from the address region.

Features used for address type classification are shown in Figure 6.

Initially, we manually constructed and tuned a set of rules for address type classification, expressed in terms of the feature values for all the address regions found on a page. The following gives the flavor of the rules employed. Address regions were first analyzed to identify those that could confidently be classified as SENDER or RECIPIENT. For example, addresses that were a single line at the bottom of a page, or appeared in the upper right hand corner of the page were classified as SENDER. When not all address regions could be classified in this way, further rules were invoked that made use of features from more than one address region. For example, when exactly two good regions are found:

```
IF (both SENDER) leave as is;
ELSE IF (one is SENDER) mark other RECIPIENT;
ELSE IF (2nd region is in BOTTOM_HALF
         AND not last block AND has >1 line)
  {
  type[2nd reg]=OTHER;
  IF (1st region is RIGHT_HALF
      OR has TEL or EMAIL)
    type[1st]=SENDER;
  ELSE type[1st]=RECIPIENT;
  }
ELSE
  {
  region with greater x0-y0 is SENDER;
  other region is RECIPIENT;
  }
```

This approach yielded moderate accuracy in discrimination, but (unsurprisingly) was difficult to tune or extend.

Subsequently, we have tested machine learning approaches to discriminating address types. Our experiments so far have addressed only the classification of individual address regions, without regard to the presence or features of other address regions found on the same page. Nonetheless, overall accuracy was somewhat better than obtained using the rule-based approach.

The best-performing classifier was a set of three linear Support Vector Machines (SVMs)[13]. Each SVM exercises a linear decision surface $\mathbf{w} \cdot \mathbf{x} + b = 0$ on feature vectors $\mathbf{x}$. For each SVM, positive values of $\mathbf{w} \cdot \mathbf{x} + b$ correspond to classification as one of the classes {SENDER, RECIPIENT, OTHER}, while negative values correspond to the other two. To classify an address region, its feature vector is submitted to each of the three SVMs; the class of the SVM having the most positive score is chosen as the address type classification. Feature vector components were normalized by linearly rescaling so that for each the training set covers the range $[-1, 1]$.

5 Implementation

Our system was implemented in C and C++ on UNIX. So far, little attention has been devoted to real time performance for address parsing opera-

tions. On a Pentium II 450 MHz processor, a typical fax business letter page image is processed in less than 15 seconds, including page orientation, OCR, and processing for address extraction.

6 Experimental Results

6.1 Performance Metrics and Truthing

Ultimately, it may be of value to locate and parse entire addresses, including person names, firm names, and other organization-related information such as titles, department names, and the like. In our initial experiments, however, we concentrated on locating and parsing addresses containing a street address or post office box plus city, state, and ZIP Code. Where they were also present, we attempted to parse telephone numbers (including fax numbers) and email addresses. Addresses could comprise one or more text blocks in geometrical layout, or could occur *inline*, e.g., as part of a textual paragraph; however, our system currently finds only addresses that lie within a single text block as determined during page layout analysis.

For fax cover sheets, we attempted in addition to extract machine printed contents of any address field contained in the same block as a `TO` or `FROM` keyword. By "block" we mean any collection of text lines that appeared to a human scorer to be part of the same logical unit. In practice, this set quite an exacting standard against which to measure system performance, since for some cover sheets the discrimination between sender and recipient address information was not immediate for human scorers.

On the other hand, we attempt to *reject* isolated occurrences of address components that don't truly constitute addresses, such as city names or telephone numbers occurring as parts of sentences.

How does one measure success in locating and parsing addresses? Clearly this depends on the application. We chose the following set of metrics:

- *Fraction of addresses found.* An address is found if any address component is returned by the system, whether or not correctly parsed.
- *Fraction of city, state, and ZIP Code components found (CSZ found).* CSZ is found if all the city, state, and postal code words are returned by the system as part of the address, whether or not correctly parsed.
- *Fraction of CSZ components correctly parsed (CSZ OK).* All CSZ words are assigned the correct part of speech during parsing.
- *Fraction of street address components found (Street found).* Street found occurs when all words of the street address component, including street number, street name, street suffix ("Avenue"), directional ("North"), and street secondary address ("Suite 200") are returned by the system as part of the address, whether or not correctly parsed.
- *Fraction of street address components correctly parsed (Street OK).* Street OK occurs when all words of the street address component are assigned the correct part of speech during parsing.

Analogous metrics are defined for the other address components (PO Box, telephone number, fax number, etc.).

6.2 Business Letters

We tested on a database of 159 standard resolution FAX images of *first pages* of U.S. business letters, from the UNLV image database [14]. These images were not used in the development of the BLADE algorithms. Results are shown in Table 5.

Table 5: Performance on UNLV Standard Resolution Fax Business Letters Database. Entries show percent correct for various tasks, separately for recipient and sender addresses. See text for explanation of row headings.

	RECIPIENT	*SENDER*
Total addresses	86	150
Address found	88%	50%
CSZ present	100%	99%
CSZ found	84%	45%
CSZ OK	83%	43%
Street present	98%	85%
Street found	69%	30%
Street OK	65%	27%

Of this set, 94% of images contain at least one address. A majority of images 79% contained exactly one sender address; 14% contained no sender address, and 8% contained more than one sender address. 54% of images contained a recipient address. 46% of images contained both a sender and a recipient address.

For recipient addresses, 88% were found; 82% had city, state, and ZIP Code correctly parsed ("CSZ OK"), and 65% had the street address correctly parsed ("Street OK") as well.

For sender addresses, 50% were found; 42% had city, state, and ZIP Code correctly parsed, and 27% had the street address correctly parsed as well.

Contributing factors to the poorer accuracy performance for sender addresses were:

- Sender addresses often appear in smaller fonts and/or italics; output from our OCR system was often of poor quality for these addresses, and they were never spotted.
- Sender addresses show a wider variation in address grammar, especially with respect to locations of line breaks.
- Sender addresses sometimes appear in a multi-column format with columns closely spaced: when such an arrangement is incorrectly segmented as a single text block during page layout analysis, the order of address components is corrupted.

Performance on fine resolution images would be expected to be somewhat better.

6.3 Fax Cover Sheets

The system for address extraction from fax cover sheets was trained and tested on a proprietary database of 177 fax cover sheet images, 138 at standard (200x100 dpi) resolution and 39 at fine (200x200 dpi) resolution. Because of the difficulty of obtaining a representative sample of fax cover sheet images, and because of the significant variability in layout among the cover sheets we collected, we decided to train our system on a subset of the test set (the first 50 images). Thus our test results may be biased toward overstating accuracy.

The training images were used in two ways:

1. To expand the portion of the grammar specific to cover sheets. When our initial grammar failed to parse some cover sheet address components, the product rules were expanded. Costs associated with the grammar were *not* trained using this image set.
2. To improve the performance of text block segmentation during page layout analysis. Initially, page layout analysis tended to split address regions (sender or recipient) across multiple text blocks, because of the relatively wide spacing between lines on fax cover sheets. We adjusted our segmentation algorithm to favor larger vertical gaps between text blocks.

We found it considerably more difficult to extract addresses from fax cover sheets than from business letters. Results are summarized in Table 6.

For the cover sheets database, 14/177 (8%) of images contained neither a sender nor a recipient address. Only 54/177 (31%) of images contained a recipient address, reflecting a large fraction of recipient address fields containing handwriting. 161/177 (91%) images contained at least one sender address region; 42/177 (24%) contained more than one.

Table 6: Performance on 177 fax cover sheets. Entries show percent correct for various tasks, separately for recipient and sender addresses. See text for explanation of row headings. CSZ and street component results are almost never present in recipient addresses.

	RECIPIENT	*SENDER*
Total addresses	54	205
Address found	24%	38%
CSZ present	6%	59%
CSZ found	6%	30%
CSZ OK	4%	22%
Street present	4%	58%
Street found	4%	29%
Street OK	2%	19%
Fax present	93%	72%
Fax found	19%	32%
Tel present	26%	71%
Tel found	9%	28%
Firm present	30%	NA
Firm found	9%	NA

52/177 (29%) of images contained both sender and recipient addresses.

Even though we started with a moderately large number of cover sheets, fewer than one third contained detectable [machine print] recipient address information. Recipient addresses were normally identified by the presence of a `TO` identifier; virtually none contained street or CSZ address components. The most frequently found recipient address component was a fax number; however, we still found this only about 20% of the time. Firm names were parsed with the aid of keyword matching on firm identifiers (e.g., "Company"); firm names were found about a third of the time when present.

The main reason for this poor performance was the difficulty in associating the recipient identifier (`TO`) with a machine printed address component that was normally in a different field. OCR errors were also a contributing factor.

Roughly 60% of sender addresses appeared in the same forms familiar from business letters – typically containing street or PO Box and CSZ components. Many occurred in "letterhead" versions of preprinted fax cover sheets. Addresses containing a CSZ component were found at rates similar to those in business letters, about 50%.

Sender address components also appeared as the contents of one or more cover sheet fields, typically consisting of a person and/or firm name, telephone number, and fax number. These were found at a lower rate, roughly 20%, due to the same difficulties as for recipient addresses.

6.4 Address Type Classification

Our address type classification subsystem was trained and tested only for fax cover sheet images.

Due to the small sample available for training, we evaluated performance by the "leave out one" procedure, where a classifier is trained on all but one sample and tested on the remaining sample (left out of training). Results are averaged over all choices of the partitioning between training and test samples. In this way, the classifier is never tested on a sample on which it has been trained, yet all samples are used for both training and testing. This approach was feasible only because classifiers could be trained rapidly on the relatively small training set.

The training/test set consisted of one feature vector (see Figure 6) for each address region yielding a valid parse, without regard to the parse score. Out of 177 fax cover sheet images, 70 yielded one or more such address regions, for a total of 97 address regions. The classifier achieved an accuracy of 79%. This compares with an accuracy of 77% for a nearest neighbor classifier. Table 7 shows the confusion matrix for the SVM classifier configuration.

Table 7: Confusion matrix for classification of address region type as {`SENDER, RECIPIENT, OTHER`} by three linear SVMs.

		Result		
TRUTH	Total	Sender	Recipient	Other
Sender	72	71	0	1
Recipient	16	10	6	0
Other	9	9	0	0

It is important to keep in mind that these results are obtained using only information about the address region being classified. Nonetheless, the overall accuracy was higher than we obtained via handcrafted rules that attempted to make use of information about *all* address regions found on a page. We would expect to see significantly improved accuracy from a hybrid approach that uses machine learning techniques in considering features from multiple address regions. Such an approach may require additional training data.

7 Conclusions

We have described a system for extracting machine printed address components from English language business letters and fax cover sheets, and have reported preliminary performance measurements.

For business letters, address extraction performance is primarily limited by OCR performance. Improvement can also be expected from expanding keyword lists, the address grammar, and the word lists used for dynamic matching of specific word types.

Performance on fax cover sheets was limited by the same factors, and others as well. Field-based address regions were often fragmented into multiple text blocks during the page layout analysis phase, often preventing recognition of an address. Techniques are needed to identify and group fields according to whether they contain sender or recipient address information.

In our preliminary tests, address type classification accuracy was mediocre. However, much potentially relevant information is not yet used. The *banner* entered at the top of each image by the sending fax machine (and often available in character form at the receiving fax machine) contains information that frequently correlates with the contents of sender address fields, and could improve type classification accuracy, though confusions are still possible. Similarly, when a cover sheet or business letter contains a signature block the person name, and company, if present, should match the sender information. A name in the salutation line of a letter usually matches the recipient name. Such features in combination with a machine learning approach that makes use of all address information found on a page should significantly improve classification performance.

References

[1] J. Schürmann, N. Bartneck, T. Bayer, J. Franke, E. Mandler, and M. Oberländer, Document analysis – From pixels to contents. *Proceedings of the IEEE* **80** (1992) 1101–1119.

[2] T. Bayer, U. Bohnacker, and I. Renz, Information extraction from paper documents, in *Handbook of Character Recognition and Document Image Analysis*, H. Bunke and P. S. P. Wang, editors (World Scientific, 1997) 653–677.

[3] S. Baumann, M. Ben Hadj Ali, A. Dengel, T. Jager, M. Malburg, A. Weigel, and C. Wenzel, Message extraction from printed documents: a complete solution, in *Proceedings of the International Conference on Document Analysis and Recognition* (IEEE Computer Society, 1997) 1055–1059.

[4] A. Dengel, R. Bleisinger, F. Fein, R. Hoch, and F. Hönes, Officemaid - a system for office mail analysis, interpretation, and delivery, in *Proceedings of the First International Workshop of Document Analysis Systems (DAS'94)*, Kaiserslautern, Germany, 1994, 253–275.

[5] C. Wenzel, Supporting information extraction from printed documents by lexico-semantic pat-

tern matching, in *Proceedings of the International Conference on Document Analysis and Recognition* (IEEE Computer Society, 1997) 732–735.

[6] H. U. Mogg-Schneider and C. Aufmuth, Information extraction from tax assessment forms, in *Document Analysis Systems II*, J. J. Hull and S. L. Taylor, editors (World Scientific, 1998) 209–222.

[7] M. Köppen, D. Waldöstl, and B. Nickolay, Information extraction from tax assessment forms, in *Document Analysis Systems II*, J. J. Hull and S. L. Taylor, editors (World Scientific, 1998) 223–241.

[8] J. Lii and S. N. Srihari, Location of name and address on fax cover pages, in *Proceedings of the International Conference on Document Analysis and Recognition* (IEEE Computer Society, 1995) 756–759.

[9] H.S. Baird, Global-to-local layout analysis, in *IAPR workshop on syntatic and structural pattern recognition* (1988) 1–16.

[10] D.-S. Lee, C. R. Nohl, and H. S. Baird, Language identification in complex, unoriented, and degraded document images, in *Proceedings of the IAPR Workshop on Document Analysis Systems*, J. J. Hull and S. L. Taylor, editors (World Scientific, 1998) 17–39.

[11] H.S. Baird, Anatomy of a versatile page reader, *Proceedings of the IEEE* **80** (1992) 1059–1065.

[12] S. Wu and U. Manber, Fast text searching allowing errors, *Communications of the ACM* **35** (October 1992) 83–91.

[13] C.J.C. Burges, A tutorial on support vector machines for pattern recognition, *Data Mining and Knowledge Discovery*, **2**(2) (1998) 121–167.

[14] S. V. Rice, F. R. Jenkins, and T. A. Nartker, The fifth annual test of ocr accuracy, Technical Report TR-96-01, Information Science Research Institute, University of Nevada, Las Vegas, April 1996.