Learning to Extract Signature and Reply Lines from Email

Vitor R. Carvalho
Language Technologies Institute
School of Computer Science
Carnegie Mellon University
vitor@cs.cmu.edu

William W. Cohen
Center for Automated Learning and Discovery
School of Computer Science
Carnegie Mellon University
wcohen@cs.cmu.edu

Abstract:

We describe methods for automatically identifying signature blocks and reply lines in plain-text email messages. This analysis has many potential applications, such as preprocessing email for text-to-speech systems; anonymization of email corpora; improving automatic content-based mail classifiers; and email threading. Our method is based on applying machine learning methods to a sequential representation of an email message, in which each email is represented as a sequence of lines, and each line is represented as a set of features. We compare several state-of-the-art sequential and non-sequential machine learning algorithms on different feature sets, and present experimental results showing that the presence of a signature block in a message can be detected with accuracy higher than 97%; that signature block lines can be identified with accuracy higher than 99%; and that signature block and reply lines can be simultaneously identified with accuracy of higher than 98%.

1. Introduction

The growing importance of email as a communication medium have inspired many attempts to develop intelligent tools for classifying, organizing, and presenting email messages (e.g., Bellotti *et al.*, 2003; Murakoshi *et al.*, 1999; Sproat *et al.*, 1998). For many of these tools, it is useful to be able to analyze the body of an email message by splitting it into components.

In this paper we present a method for component-level analysis of plain-text email messages, and use it to identify two important types of components: the *signature block* and the *reply lines* of messages. Identification of these components has many potential applications, including preprocessing email for text-to-speech systems (Sproat *et al.*, 1998); automatic formation of personal address lists; and anonymization of email corpora. Our own interest stems from the need to preprocess email messages in order to classify email according to speech acts (Cohen *et al.*, 2004).

Component-level analysis of email bodies is difficult, since the language used in email tends to be diverse, informal, and (particularly for signature blocks) often even creative. We use supervised learning methods on a corpus of annotated messages to develop component-level analysis tools. We first consider the *detection* of signature blocks—that is, determining if an email contains a signature. We then consider identifying the actual lines of an email which comprise the signature block. For this task, we represent every email message as a sequence of lines, with each line represented by a set of features. Finally we extend the signature block line identification method to identify *reply lines*—i.e., lines quoted from an earlier message in an email thread. For each of these subtasks, we show that high levels of accuracy (in the high 90's) can be obtained.

In our experiments we compare several learning algorithms, including some recently-developed *sequential learning* techniques such as Conditional Random Fields, or CRFs (Lafferty *et al.*, 2001), Conditional Markov Models, or CMMs (McCallum *et al.*, 2000; Klein *et al.*, 2002), and a method for discriminatively training Hidden Markov Models (HMMs) using voted perceptrons (Collins, 2002). These are compared to their non-sequential analogs, as well as other well-studied non-sequential learners such as boosted decision trees (Schapire *et al.*, 1998) and Naïve Bayes. Thus another contribution of the paper is a rigorous comparative evaluation of these methods on two real-world problems.

2. Problem Definition and Corpus

A signature block is the set of lines, usually in the end of a message, that contain information about the sender, such as personal name, affiliation, postal address, web address, email address, telephone number, etc. Quotes from famous persons and creative ASCII drawings are often present in this block, also. An example of a *signature block* can be seen in last six lines of the email message pictured in Figure 1 (marked with the line label <sig>). Figure 1 also contains six lines of text that were quoted from a preceding message (marked with the line label <reply>). In this paper we will call such lines *reply lines.*

```
<other>  From: wcohen@cs.cmu.edu
<other>  To: Vitor Carvalho <vitor@cs.cmu.edu>
<other>  Subject: Re: Did you try to compile javadoc recently?
<other>  Date: 25 Mar 2004 12:05:51 -0500
<other>
<other>  Try cvs update -dP, this removes files & directories that have been
deleted from cvs.
<other>  - W
<other>
<reply>  On Wed, 2004-03-24 at 19:58, Vitor Carvalho wrote:
<reply>  > I just checked-out the baseline m3 code and
<reply>  > "Ant dist" is working fine, but "ant javadoc" is not.
<reply>  > Thanks
<reply>  > Vitor
<other>
<sig>    -----------------------------------------------------------------
<sig>    William W. Cohen                       "Would you drive a mime
<sig>    wcohen@cs.cmu.edu                          nuts if you played an
<sig>    http://www.wcohen.com                          audio tape at full
<sig>    Associate Research Professor                       blast?" ----
<sig>    CALD, Carnegie-Mellon University                       S. Wright
```

Figure 1 - Excerpt from a labeled email message

Below we first consider the task of *detecting* signature blocks—that is, classifying messages as to whether or not they contain a signature block. We next consider *signature line extraction*. This is the task of classifying lines within a message as to whether or not they belong to a signature block. In our experiments, we perform signature line extraction only on messages which are known to contain a signature block.

To obtain a corpus of messages for signature block detection, we began with messages from the 20 Newsgroups dataset (Lang, 1995). We began by separating the messages into two groups *P* and *N*, using the following heuristic. We first looked for pairs of messages from the same sender and whose last T lines were identical. If T was larger or equal to 6, then one of the messages from this sender (randomly chosen) was placed in group *P* (which contains messages likely to have a signature block). If T was less than or equal to 1, a sample message from this sender was placed in group *N*. These groups were supplemented with messages from our personal inboxes (to provide a sample of more recent emails) and manually checked for correctness. This resulted in a final set of 617 messages (all from different senders) containing a signature block, and a set of 586 messages not having a signature block.

For the extraction experiments, the 617-message dataset was manually annotated for signature lines. It was also annotated for *reply* lines (as in Figure 1). As noted above, the identification of reply lines can be helpful in tasks such as email threading, and certain types of content-based message classification; and as we will demonstrate below, our signature line extraction techniques can also be successfully applied to identifying reply lines. The final dataset has 33,013 lines. Of these, 3,321 lines are in signature blocks, and 5,587 are reply lines.

3. Signature Block Detection

In signature block detection, an email message is represented by a set of features, and a classifier is learned over this feature space. The features we used for the detection problem are summarized in Table 1. Each of these *feature patterns* is applied to each one of the last K lines of the email message; for instance, one feature generated by the second pattern of the table might be "a URL appears in the 3-rd from last line". All regular expressions follow Java syntax.

Feature Pattern
Line contains email pattern
Line contains URL pattern
Line contains phone number pattern
Line matches the regular expression "^[\s]*---*[\s]*$"
Line has a sequence of 10 or more special characters, as in the following regular expression: "^[\s]*([*]|#|[\+]|[\^]|-|[\~]|[\&]|[///]|[\$]|_|[\!]|[\/]|[\%]|[\:]|[\=]){10,}[\s]*$"
Line contains any these typical signature words: "Dept\.|University|Corp\.|Corporations?|College|Ave\.|Laboratory|[D|d]isclaimer| Division|Professor|Laboratories|Institutes?|Services|Engineering|Director|Sciences?| Address|Manager|Street|St\.|Avenue"
Line contains a pattern like Vitor R. Carvalho or William W. Cohen, as in regular expression: "[A-Z][a-z]+\s\s?[A-Z][\.]?\s\s?[A-Z][a-z]+"
Line ends with quote symbol, as in regular expression: "\"$"
Line contains the name of the message sender, Surname, or Both (If it can be extracted from the message header)
The number of leading tabs in the line (as in regular expression "\t") equals 1
The number of leading tabs equals 2
The number of leading tabs is equal or greater than 3
Percentage of punctuation symbols (as in regular expression "\p{Punct}") in the line is larger than 20%
Percentage of punctuation symbols in the line is larger than 50%
Percentage of punctuation symbols in the line is larger than 90%

Table 1 - Features Used in the *Signature Block* Detection Task

Since feature patterns are applied to each of the last K lines of the email message, the total number of features representing a message grows with K; for instance, if the URL pattern feature is found in each of the last 3 lines of a message, three different features will be present in the representation of the email. Restricting these features to the last K lines is thus a type of feature selection, since signature blocks are generally in the last few lines of a message.

With the above features, the signature block detection problem was thus reduced to a binary classification problem. Using a 5-fold cross-validation procedure on the 1,203 labeled messages, we obtained the results of Table 2. Here F1 is the harmonic precision-recall mean, defined as (2×Precision×Recall)/(Recall+Precision).

SIGN HERE

Learning Algorithm	K = 5			K = 10			K = 15		
	F1	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall
Naïve Bayes	89.67	81.81	99.18	87.31	77.58	99.83	83.49	71.66	100
Maximum Entropy	95.11	97.28	93.03	97.40	97.56	97.24	96.98	97.54	96.43
SVM	94.87	96.79	93.03	**97.55**	98.03	97.08	**97.39**	97.87	96.92
VotedPerceptron	**95.19**	97.45	93.03	96.39	97.35	95.46	95.59	96.22	94.97
AdaBoost	95.16	96.19	94.16	96.76	96.45	97.08	96.56	97.36	95.78

Table 2 - Detecting Signatures in Emails: Results

Regarding the classifiers, SVM is the support vector machine algorithm with a linear kernel (Joachims, 2001). Voted Perceptron is the algorithm described in (Freund *et al.*, 1999), trained using a single pass through the data. AdaBoost is an implementation of the confidence-rated boosting method described in (Schapire *et al.*, 1999), in which a weak learner (in our case, a depth-5 decision tree) is boosted 10 times. Maximum Entropy (a.k.a. logistic regression) uses limited-memory quasi-Newton optimization (Sha and Pereira, 2003) and a Gaussian prior.

Overall, the best results of detection (F1=97.55) are found by using SVM and K=10, i.e., only the last 10 lines of the message. However, all classifiers other than Naïve Bayes perform well. Most of the mistakes occurred in signature blocks containing either only an ASCII drawing, only the nickname of the sender, or only a few quoted sentences.

For all classifiers (other than Naïve Bayes), the best performances were reached with K=10, and for K=15, there is a slight decrease in performance. Interestingly, Sproat *et al.* (1999) also observed that "SIG fields are rarely longer than ten lines".

4. Signature Line Extraction

For the task of extracting signature block lines, we represented each email document not as a set of features, but as a sequence of lines. Each line is labeled as to whether or not it is part of a signature block, and represented as a set of features, similar to the features described above. This approach thus reduces the signature line extraction problem to a sequential classification problem. Sequential classification problems can be addressed with sequential learning methods (like HMMs or CRFs) or else the ordering of the lines can be ignored, and conventional non-sequential learning methods can be applied. We also observed improvements on this task by representing lines not only with its own features, but also with the features derived from neighboring lines.

The complete list of features used in this extraction problem can be seen in Table 4. The first column of Table 4 describes the features, and the next 3 columns ("on current line", "on previous line" and "on next line"), indicate to which lines it should be applied. For instance, in order to represent line number 25 of an email message as a set of features, the *Blank Line* feature is checked not only in line 25, but also in line 24 and 26. In a message in which these 3 lines are indeed blank lines, the representation of line number 25 will contain the following 3 features: *blankLine*, *prevBlankLine* and *nextBlankLine*.

Many of these features are "overlapping" or non-independent. We also note that some of the features described in Table 4 were originally created to extract *reply* lines (e.g., the typical reply marker); however, they proved to be useful for signature line extraction as well.

Results using the features from Table 4 are shown in Table 3. They were obtained using 5-fold cross-validation procedure on the previously mentioned set of 617 labeled messages. (The cross-validation process was constrained so that lines from the same message were never split between training and test sets.) Of the 33,013 lines, 3,321 are signature block lines.

Learning Algorithm	**Without Features from Previous and Next Lines**				**With Features from Previous and Next Lines**			
	Accuracy (%)	F1	Precision	Recall	Accuracy (%)	F1	Precision	Recall
Non-Sequential								
Naïve Bayes	94.13	73.88	66.80	82.65	91.03	68.60	52.95	97.38
Maximum Entropy	96.26	80.16	86.07	75.00	99.11	95.56	96.38	94.76
SVM	96.41	80.39	89.41	73.02	**99.12**	**95.62**	96.10	95.15
VotedPerceptron	96.10	80.23	81.88	78.65	98.96	94.73	96.32	93.19
AdaBoost	**96.53**	**82.12**	85.44	79.04	99.11	95.55	96.21	94.91
Sequential								
CPerceptron(5, 25)	97.01	83.62	93.02	75.94	**99.37**	**96.82**	98.20	95.48
CMM(SVM, 5)	91.28	66.82	54.11	87.35	99.27	96.37	97.21	95.54
CRF	**98.13**	**90.97**	88.05	94.09	99.17	95.97	94.27	97.74

Table 3 - Signature Line Extraction: Results

The non-sequential learners are the same used for the detection problem in Section 3. The CMM(SVM, 5) sequential learner is a type of conditional Markov model (Klein *et al.*, 2002); briefly, each example of the SVM is extended with the predicted classes of the previous 5 lines, and a beam search is used to find the sequence of predictions that maximize total prediction confidence, where total prediction confidence is the sum of the distance of each prediction from the separating hyperplane. An SVM with linear kernel is used as the base classifier. The last algorithm, CRF, is an implementation of conditional random fields (Lafferty *et al.,* 2001). As in our implementation of maximum entropy, limited-memory quasi-Newton optimization and Gaussian priors were used.

CPerceptron(5,y) is an implementation of the HMM-learning algorithm proposed by Collins (2002), also with a history of 5 previous classifications. The "y" value is the number of passes over the training data that were made. In order to determine the optimum value for "y", we used a development set with approximately 10% of the original dataset size and searched for the best "y" value between 1 and 40. For the signature block dataset, the best number of iterations was found to be 25. For the problems in Section 5 and 6, the best "y" values were found to be 18 and 38, respectively.

Line Features Description	On current line	On previous line	On next line
Blank line	X	X	X
Email pattern	X	X	X
Last Line	X		
Previous to last Line	X		
Email Header pattern	X		
Email pattern	X	X	X
URL pattern	X	X	X
Phone number pattern	X		
This signature marker regular expression: "^[\s]*---*[\s]*$"	X	X	X
A line with a sequence of 10 or more special characters, as in the following regular expression: "^[\s]*([*]|#|[\+]|[\^]|-|[\~]|[\&]|[///]|[\$]|_|[\!]|[\/]|[\%]|[\:]|[\=]){10,}[\s]*$"	X	X	X
The presence of any these typical signature words:"Dept\.|University|Corp\.|Corporations?|College|Ave\.|Laboratory|[D|d]isclaimer|Division|Professor|Laboratories|Institutes?|Services|Engineering|Director|Sciences?| Address|Manager|Street|St\.|Avenue"	X	X	X
Names patterns like Vitor R. Carvalho or William W. Cohen, as in regular expression: "[A-Z][a-z]+\s\s?[A-Z][\.]?\s\s?[A-Z][a-z]+"	X		
Lines ending with quote symbol, as in regular expression: "\"$"	X		
The Name of the email sender, Surname, or Both (If it can be extracted from the email header)	X		
The number of tabs (as in regular expression "\t") equals 1	X	X	X
The number of tabs equals 2	X	X	X
The number of tabs is equal or greater than 3	X	X	X
Percentage of punctuation symbols (as in regular expression "\p{Punct}") is larger than 20%	X	X	X
Percentage of punctuation symbols in a line is larger than 50%	X	X	X
Percentage of punctuation symbols in a line is larger than 90%	X	X	X
Typical reply marker (as in regular expression "^\>")	X	X	X
Line starts with a punctuation symbol	X	X	X
Next line begins with same punctuation symbol as current line	X		
Previous line begins with same punctuation symbol as current line	X		
Line starts with 1 or 2 punctuation symbols, which are followed by the typical reply marker, as in regular expression: "^\p{Punct}{1,2}\>"	X	X	X
Reply line clue line endings, as in regular expression: " wrote:$" or " writes:$",	X	X	X
Percentage of letters or numbers (as in regular expression "\[a-zA-Z0-9]") is smaller than 90%	X	X	X
Percentage of letters or numbers in a line is smaller than 50%	X	X	X
Percentage of letters or numbers in a line is smaller than 10%	X	X	X

Table 4 - Complete List of Features used for Line Extraction

For the non-sequential learners, the features of the neighboring lines (henceforth *window features*) give a huge improvement in performance: the best F1 scores grow from 82.12 without window features (for AdaBoost) to 95.62 (for SVM). The sequential learners also benefit from the window features, but by a smaller margin. CRF performs best (F1=90.97) when window features are not used, and CPerceptron performs best when window features are used (F1= 96.82).

In general, the sequential learners perform better than the non-sequential ones. For this task, CRF is by far the best method without the window features. With the window features, CPerceptron(5,25) is the best performer (and also the best overall, with the impressive accuracy of 99.37%); however, CMM(SVM, 5) is a close second.

It is interesting to note that, on this problem, the "feature engineering" step of adding *window features* affects performance far more the adoption of sequential learners. When the *window features* are used, the difference in performance between the sequential and non-sequential learners is fairly small.

5. Reply Line Extraction

The learning methods above can be applied to other types of email body analysis. We considered as a second task the goal of identifying reply lines in an email. The same representation of email messages was used—as a sequence of lines, each of which is a set of features—and the same set of features (described in Table 4) was used to describe each line. Once again, this leads to a sequential binary classification problem.

The results shown in Table 5 were again obtained using 5-fold cross-validation procedure on the labeled set of 617 messages. Of the 33,013 lines, a total of 5,587 are reply lines.

Learning Algorithm		**Without Features from Previous and Next Lines**					**With Features from Previous and Next Lines**			
		Accuracy (%)	F1	Precision	Recall		Accuracy (%)	F1	Precision	Recall
Non-Sequential										
Starts with ">"		95.10	83.08	99.92	71.09		n/a			
Naïve Bayes		97.97	93.98	94.47	93.50		93.86	84.37	74.03	98.06
MaximumEntropy		98.23	94.57	98.11	91.28		98.74	96.22	97.64	94.84
SVM		98.32	94.90	97.96	92.03		**98.83**	**96.52**	97.25	95.81
VotedPerceptron		98.19	94.38	99.19	90.03		98.48	95.36	98.90	92.07
AdaBoost		**98.46**	**95.33**	97.77	93.00		98.73	96.20	96.72	95.68
Sequential										
CPerceptron(5, 18)		98.05	94.19	95.32	93.09		98.73	96.20	97.62	94.82
CMM(SVM, 5)		97.80	93.36	95.06	91.73		98.66	95.89	98.89	93.07
CRF		**98.10**	**94.31**	95.55	93.10		**99.04**	**97.15**	98.17	96.15

Table 5 - Reply Lines Extraction: Results

This problem would seem to be easier than extracting signature lines. This is confirmed by the fact that the simple baseline of simply looking for lines that start with an angle bracket (">") performs well (F1=83.08). There is a benefit in using the window features, but much less significant than the one observed for the signature block line extraction problem. Interestingly, when the window features are *not* utilized, all non-sequential learners (with the exception of Naïve Bayes) outperform the corresponding sequential ones.

For the reply lines identification task, CRF was the most successful algorithm overall. Using the window features, it could correctly predict more than 99% of the 33,013 lines.

6. Multi-Class Line Identification

In a final experiment, we considered the task of extracting from an email both reply lines and *signature block* lines. We treated this as a multi-class sequential classification problem, in which each line is given one of three labels: *signature, reply,* or *other*. We again used the features of Table 4.

The results shown in Table 6 were obtained by using 5-fold cross-validation on the set of 617 manually labeled messages. The performance is measured by accuracy over all lines, and we also give for each learner a confusion-matrix (where rows indicate the true class, and the columns indicate the class predicted by the learning algorithm). All confusion-matrix entries are percentages of the total set of lines.

Multi-class Sequential Learning Algorithm	**Without Features from Previous and Next Lines**					**With Features from Previous and Next Lines**				
	Accuracy (%)	Confusion-Matrix				Accuracy (%)	Confusion-Matrix			
		%	Sig	Reply	Other		%	Sig	Reply	Other
CPerceptron(5, 38)	95.35	Sig	**8.27**	0.17	1.61	**98.91**	Sig	**9.85**	0.06	0.15
		Reply	0.05	**15.22**	1.65		Reply	0.14	**16.39**	0.38
		Other	0.37	0.78	**71.85**		Other	0.09	0.26	**72.65**
		%	Sig	Reply	Other		%	Sig	Reply	Other
CRF	**96.71**	Sig	**9.42**	0.03	0.61	98.48	Sig	**9.85**	0.05	0.16
		Reply	0.04	**15.87**	1.00		Reply	0.06	**16.32**	0.54
		Other	1.36	0.24	**71.41**		Other	0.51	0.20	**72.30**

Table 6 - Results for Simultaneous Signature Block and Reply Lines Extraction

Once again, all learning methods benefit from the window features, specially the CPerceptron. The CPerceptron algorithm outperforms CRF when the window features are used, reaching an accuracy (or percentage of correctly labeled lines) of 98.91%. CRF is the best learner when no window features are used, correctly labeling 96.71% of all lines.

7. Conclusion

In this work we addressed the problem of identifying components within the body of an email, specifically *signature blocks* and *reply lines*. We first considered attempting to detect whether or not an email message contained a signature block. We developed a set of linguistically inspired features, which when applied to the last 10 lines of the email messages are highly informative: experiments with several learning algorithms produced highly accurate classifiers, with F1 measures above 97.5%.

We then considered the task of extracting the specific lines of an email that comprised a signature. For this "line extraction" problem, a message was represented as a sequence of lines, with each line represented as a set of features collected not only from the current line, but also from the previous and next lines. The addition of these "window features" proved to boost the extraction performance considerably. In experiments with sequential and non-sequential learning algorithms, high accuracy (above 99%) and F1 measures (nearly 97%) were achieved. For this problem, the choice of learning algorithm and feature set is much more important: for example, without window features, the best non-sequential learner obtains an F1 measure of only 82.1%.

We also used the same methods to extract *reply* lines of email messages, again obtaining accuracy measures above 99%. Finally, we used multi-class sequential learning techniques to extract both reply and signature block lines simultaneously. Again very good results were achieved, with accuracy of 98.91% for the task.

In prior work using machine learning algorithms for different line-classification tasks, Pinto *et al.* (2003) study the problem of identifying lines of tables, and McCallum *et al.* (2000) present results on extracting lines of "frequently asked questions" (FAQ) documents. From the prospective of machine learning research, our experiments provide an additional comparative study of sequential learning methods, applied to a new set of line-classification problems.

In prior work on email body analysis, Chen *et al.* (1999) describe a method to parse signature block fields in email messages using a combination of two-dimensional structural segmentation and one-dimensional grammatical constraints, integrated with weighted finite-state transducers. (A similar combination of layout and linguistic information was also used by Hurst *et al.* (2000) to analyze general tabular data.) Chen *et al.* (1999) report a 93% recall rate and a 90% precision rate for identifying signature blocks among all blocks in a message, given that the blocks were previously separated. These results cannot be directly compared with ours, as we consider line classification, and not block identification. We also do not use any features based on the lay-out of the email messages.

As a practical matter, our method is likely to be much more easily reproduced than that of Chen *et al. (1999).* In particular, we have presented a highly accurate analysis technique which relies only on a set of easy-to-implement features, computed on individual lines of an email, which we have completely described in this paper. Using these features, the best performance is usually obtained with one of two sequential learning methods—CRFs (Lafferty *et al.,* 2000) or CPerceptron, a method of discriminatively training HMMs with voted perceptrons due to Collins (2002). Of these two methods, we note that CPerceptron has the virtue of being quite easy to implement. We also note that with the best feature sets described, very good performance can also be obtained with widely available non-sequential learning methods.

8. References

V. Bellotti, N. Ducheneaut, M. Howard and I. Smith. *Taking email to task: the design and evaluation of a task management centered email tool.* Proc. of the Conference on Human Factors in Computing Systems, Ft. Lauderdale, Florida.2003

W. W. Cohen, V. R. Carvalho and T. M. Mitchell. *Learning to Classify Email into Speech Acts.* Submitted for publication. 2004

H. Chen, J. Hu, and R. Sproat. *Integrating geometrical and linguistic analysis for e-mail signature block parsing.* ACM Transactions on Information Systems, 17(4):343--366, October 1999.

M. Collins. *Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms*, Proceedings of the EMNLP 2002, 2002

Y. Freund and R. Schapire. Large Margin Classification Using the Perceptron Algorithm. Machine Learning 37(3), 1999.

M. Hurst and T. Nasukawa. *Layout and Language: Integrating Spatial and Linguistic Knowledge for Layout Understanding Tasks.* Proceedings of the COLING 2000. 2000

T. Joachims. *A Statistical Learning Model of Text Classification with Support Vector Machines.* Proc. of the Conference on Research and Development in Information Retrieval (SIGIR), ACM, 2001.

D. Klein and C. D. Manning. *Conditional Structure versus Conditional Estimation in NLP Models.* 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2002

J. Lafferty, A. McCallum and F. Pereira. *Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data.* Proceedings of the ICML-2001. 2001

A. McCallum, D. Freitag and F. Pereira. *Maximum Entropy Markov Models for Information Extraction and Segmentation.* Proceedings of the ICML-2000, 2000

H. Murakoshi, A. Shimazu and K. Ochimizu. *Construction of Deliberation Structure in Email Communication.* Pacific Association for Computational Linguistics, pp. 16-28, Waterloo, Canada, 1999

D. Pinto, A. McCallum, X. Wei and W. B. Croft. *Table Extraction Using Conditional Random Fields*, SIGIR, ACM, Toronto, Canada, 2003

R. E. Schapire and Y. Singer. *Improved boosting algorithms using confidence-rated predictions.* The 11th Annual Conference on Computational Learning Theory, Madison, WI. 1998

F. Sha and F. Pereira. *Shallow Parsing with Conditional Random Fields*. HLT-NAACL, ACM, 2003

R. Sproat, J. Hu, and H. Chen. *Emu: An e-mail preprocessor for text-to-speech.* In 1998 Workshop on Multimedia Signal Processing, pages 239--244, Redondo Beach, CA, December 1998.