Chinese Word Segmentation with Conditional Random Fields and Integrated Domain Knowledge

Andrew McCallum
Computer Science Department
University of Massachusetts Amherst
Amherst, MA 01002
mccallum@cs.umass.edu

Fang-fang Feng
Computer Science Department
University of Massachusetts Amherst
Amherst, MA 01002
fang@cs.umass.edu

Abstract

Chinese word segmentation is a difficult, important and widely-studied sequence modeling problem. Conditional Random Fields (CRFs) are a new discriminative sequence modeling technique that supports the incorporation of many rich features. This paper demonstrates the ability of CRFs to easily integrate domain knowledge, and thus reduce the need for labeled data. Using readily-available domain knowledge in the CRF's feature definitions we show that, even training on as few as 140 segmented Chinese sentences, we can achieve world-class segmentation accuracy. Furthermore, when training on more data, our approach yields segmentation F1 of 97.5% with the Penn Chinese Treebank, and realistic, incomplete lexicons—a 50% reduction in error compared with the previously best published results of which we are aware. We also introduce two alternative prior distributions for CRF's learned weights.

1 Introduction and Related Work

Unlike western languages, Chinese is not written with spaces separating words. Some words consist of a single Chinese character, but many words consist of two, three, four or more characters. A human reader of Chinese will naturally segment the character stream into words in order to discern its meaning. It is generally accepted, therefore, that a necessary first step in most Chinese language technology applications (including document retrieval, summarization and extraction) is to segment the character stream into words.

The problem of Chinese word segmentation has been widely studied, and approached from several different angles. Some researchers have used a large lexicon of Chinese words and simple rules to select among several different segmentations consistent with the lexicon (Chen and Liu, 1992; Wu and Tseng, 1993). However these approaches have been limited by the impossibility of creating a lexicon that includes all possible Chinese words and by the lack of statistical sophistication in the rules.

Others have used unsupervised learning techniques and strong statistical techniques to train on a large body of unsegmented Chinese text (Ando and Lee, 2000). Some also incorporate lexicons (Ponte and Croft, 1996; Peng and Schuurmans, 2001). They successfully cluster characters sequences into words by using the relative predictabilty of the next character. However, the lack of supervision fundamentally prevents this approach from being able to separate the words of new highly correlated word sequences such as idioms and common phrases.

In response, others have taken advantage of the availabilty of human-segmented Chinese corpora, and used supervised machine learning methods—including techniques such as finite state transducers (Sproat et al., 1996), hidden Markov models (Teahan et al., 2000), maximum entropy classifiers and transformation-based learning (Xue and Converse, 2002). However, the amount of labeled training data

will always be limited, and given the large number of parameters that must be learned in these models, performance degredation due to over-fitting is often significant. This over-fitting could be ameliorated by use of domain knowledge to provide prior biases, but models that combine the sophistication of finite state decoding with the ability to integrate domain knowledge have previously been elusive.

Conditional Random Fields (CRFs) (Lafferty et al., 2001) are recent models that have the ability to combine rich domain knowledge, with finite-state decoding, sophisticated statistical methods, and discriminative, supervised training. In their most general form, they are arbitrary undirected graphical models trained to maximize the conditional probability of the desired outputs given the corresponding inputs. In the special case we use here, they can be roughly understood as discriminatively-trained hidden Markov models with next-state transition functions represented by exponential models (as in maximum entropy classifiers), and with great flexibilty to view the observation sequence in terms of arbitrary, overlapping features, including long-range dependencies, and multiple levels of grainularity.

This paper demonstrates the success of CRFs on Chinese word segmentation, with special focus on reducing the need for labeled training data by injecting domain knowledge into the model. We believe that machine learning models should not be trained *tabula rasa*—instead, we should use models that will let their users easily integrate background knowledge whenever it is available.

To apply CRFs to Chinese word segmentation, we first cast segmentation as a sequence labeling problem: each character that begins a word is labeled with the START tag, every other character is labeled with the NONSTART tag. The task of a trained CRF is to recover the sequence of tags given an unlabeled (unsegmented) sentence of Chinese characters. This is performed by constructing a CRF whose states are each associated with a tag, then running an analogue of the Viterbi algorithm to efficiently find the most likely state sequence given the input character sequence. Finally we then output the tags associated with the states in that state sequence. We incorporate strong domain knowledge into our model by making use of 28 different special-purpose lexicons of Chinese words and characters that are readily available from the Internet and other sources. The various lexicons include common surnames, country names, location names, job titles, punctuation characters, digits, and characters that indicate the word endings of adjectives and adverbs. The features used by the CRF are conjunctions of position-shifted membership queries of these lexicons.

Comparing Chinese word segmentation accuracies can be difficult because many papers use different data sets and ground-rules. Some published results claim 98% or 99% segmentation precision and recall, *e.g.* (Chen and Liu, 1992), but these either count only the testing words that occur in the lexicon, or use unrealistic lexicons that have extremely small or artificially non-existant out-of-vocabulary rates, or use short sentences with many numbers. On the more difficult Penn Chinese Treebank data, the best performance of which we are aware is F1 of 95.2% (Xue and Converse, 2002). In correspondingly careful experiments with a set-aside test-set, we have obtained F1 of 97.5% on Chinese Treebank, cutting error nearly by half. In our test set, 5% of the words are out-of-vocabulary; of these the CRF segments 87% correctly. Notably, when trained with only 140 sentences, CRFs still reach F1 95.7%. A CRF trained with character features instead of the domain-knowledge-laden lexicon features obtains only F1 92.6% on the same amount of training data.

Pointers to publicly-available Java source code for training and testing, our lexicons, and an easily deployed pre-trained segmenter will be provided in the final copy of this paper.

This paper also presents two alternate priors distributions for the learned weights of CRFs, designed to form "sparser" solutions than the traditional Gaussian prior. However, we find that these alternatives do not improve the performance of Chinese word segmentation.

2 Conditional Random Fields

Conditional Random Fields (CRFs) (Lafferty et al., 2001) are undirected graphical models used to calculate the conditional probability of values on designated output nodes given values on designed input nodes.

In the special case in which the designated output

nodes of the graphical model are linked by edges in a *linear chain*, CRFs make a first-order Markov independence assumption among output nodes, and thus correspond to finite state machines (FSMs). Thus, in this case CRFs can be roughly understood as conditionally-trained hidden Markov models. CRFs of this type are a globally-normalized extension to *Maximum Entropy Markov Models* (MEMMs) (McCallum et al., 2000) that avoid the *label-bias problem* (Lafferty et al., 2001). Although the details of CRFs have been introduced elsewhere, we present them here in moderate detail for the sake of clarity, and because we use a different training procedure.

Let $\mathbf{o} = \langle o_1, o_2, ...o_T \rangle$ be some observed input data sequence, such as a sequence of Chinese characters, (the values on n input nodes of the graphical model). Let $\mathcal{S}$ be a set of FSM states, each of which is associated with a label, $l \in \mathcal{L}$, (such as a label NOTSTART). Let $\mathbf{s} = \langle s_1, s_2, ...s_T \rangle$ be some sequence of states, (the values on T output nodes). CRFs define the conditional probability of a state sequence given an input sequence as

$$p_\Lambda(\mathbf{s}|\mathbf{o}) = \frac{1}{Z_\mathbf{o}} \exp\left(\sum_{t=1}^{T}\sum_{k} \lambda_k f_k(s_{t-1}, s_t, \mathbf{o}, t)\right),$$

where $Z_\mathbf{o}$ is a normalization factor over all state sequences, $f_k(s_{t-1}, s_t, \mathbf{o}, t)$ is an arbitrary feature function over its arguments, and λ_k is a learned weight for each feature function. A feature function may, for example, be defined to have value 0 in most cases, and have value 1 if and only if s_{t-1} is state #1 (which may have label NOTSTART), and s_t is state #2 (which may have label START), and the observation at position t in $\mathbf{o}$ is a Chinese character appearing in the lexicon of surnames. Higher λ weights make their corresponding FSM transitions more likely, so the weight λ_k in this example should be positive since Chinese surname characters often occur at the beginning of a word. More generally, feature functions can ask powerfully arbitrary questions about the input sequence.

CRFs define the conditional probability of a label sequence based on total probability over the state sequences,

$$p_\Lambda(\mathbf{l}|\mathbf{o}) = \sum_{\mathbf{s}:l(\mathbf{s})=\mathbf{l}} p_\Lambda(\mathbf{s}|\mathbf{o}),$$

where $l(\mathbf{s})$ is the sequence of labels corresponding to the labels of the states in sequence $\mathbf{s}$.

Note that the normalization factor, $Z_\mathbf{o}$, (also known in statistical physics as the *partition function*) is the sum of the "scores" of all possible state sequences,

$$Z_\mathbf{o} = \sum_{s \in \mathcal{S}^T} \exp\left(\sum_{t=1}^{T}\sum_{k} \lambda_k f_k(s_{t-1}, s_t, \mathbf{o}, t)\right),$$

and that the number of state sequences is exponential in the input sequence length, T. In arbitrarily-structured CRFs, calculating the partition function in closed form is intractable, and approximation methods such as Gibbs sampling or loopy belief propagation must be used. In linear-chain-structured CRFs, as we have here for sequence modeling, the partition function can be calcuated very efficiently by dynamic programming. The details are given in the following subsection.

2.1 Efficient Inference in CRFs

As in the *forward-backward* for hidden Markov models (HMMs) (Rabiner, 1990), the probability that a particular transition was taken between two CRF states at a particular position in the input sequence can be calculated efficiently by dynamic programming. We can define slightly modified "forward values", $\alpha_t(s_i)$, to be the probability of arriving in state s_i *given* the observations $\langle o_1, ...o_t \rangle$. We set $\alpha_0(s)$ equal to the probability of starting in each state s, and recurse:

$$\alpha_{t+1}(s) = \sum_{s'} \alpha_t(s') \exp\left(\sum_{k} \lambda_k f_k(s', s, \mathbf{o}, t)\right).$$

The backward procedure and the remaining details of Baum-Welsh are defined similarly. $Z_\mathbf{o}$ is then $\sum_s \alpha_T(s)$. The Viterbi algorithm for finding the most likely state sequence given the observation sequence can be correspondingly modified from its original HMM form.

2.2 Training CRFs

The λ weights of a CRF are typically set to maximize conditional log-likelihood, L, of labeled sequences in some training set, $\mathcal{D} =$

$\{\langle \mathbf{o}, \mathbf{l} \rangle^{(1)}, ... \langle \mathbf{o}, \mathbf{l} \rangle^{(j)}, ... \langle \mathbf{o}, \mathbf{l} \rangle^{(N)}\}$:

$$L = \sum_{j=1}^{N} \log \left(p_\Lambda(\mathbf{l}^{(j)} | \mathbf{o}^{(j)}) \right) - \sum_k \frac{\lambda_k^2}{2\sigma}$$

where the second sum is a Gaussian prior over parameters, with variance σ, that provides smoothing to help cope with sparsity in the training data (Chen and Rosenfeld, 1999).

When the training labels make the state sequence unambiguous (as they often do in practice), the likelihood function in exponential models such as CRFs is convex, so there are no local maxima, and thus finding the global optimum is guaranteed. It is not, however, straightforward to find it quickly. Parameter estimation in CRFs requires an iterative procedure, and some methods require fewer iterations than others.

Although the original presentation of CRFs (Lafferty et al., 2001) described training procedures based on iterative scaling (Lafferty et al., 2001), it is significantly faster to train CRFs and other "maximum entropy"-style exponential models by a quasi-Newton method, such as L-BFGS (Byrd et al., 1994; Malouf, 2002; Sha and Pereira, 2002). This method approximates the second-derivative of the likelihood by keeping a running, finite window of previous first-derivatives. Sha and Pereira (2002) show that training CRFs by L-BFGS is several orders of magnitude faster than iterative scaling, and also significantly faster than conjugate gradient.

L-BFGS can simply be treated as a black-box optimization procedure, requiring only that one provide the first-derivative of the function to be optimized. Assuming that the training labels make the state paths unambiguous, let $\mathbf{s}^{(j)}$ denote that path, then the first-derivative of the log-likelihood is

$$\frac{\delta L}{\delta \lambda_k} = \left(\sum_{j=1}^{N} C_k(\mathbf{s}^{(j)}, \mathbf{o}^{(j)}) \right) - \left(\sum_{j=1}^{N} \sum_{\mathbf{s}} p_\Lambda(\mathbf{s}|\mathbf{o}^{(j)}) C_k(\mathbf{s}, \mathbf{o}^{(j)}) \right) - \frac{\lambda_k}{\sigma}$$

where $C_k(\mathbf{s}, \mathbf{o})$ is the "count" for feature k given $\mathbf{s}$ and $\mathbf{o}$, equal to the sum of $f_k(s_{t-1}, s_t, \mathbf{o}, t)$ values for all positions, t, in the sequence $\mathbf{s}$. The last term, λ_k/σ, is the derivative of the Gaussian prior.

The upper parenthesized term corresponds to the expected count of feature k *given that the training labels are used to determine the correct state paths.* The lower parenthesized term corresponds to the expected count of feature k *using the current CRF parameters, Λ, to determine the likely state paths.* Matching simple intuition, notice that when the state paths chosen by the CRF parameters match the state paths from the labeled data, the derivative will be 0.

When the training labels do not disambiguate a single state path, expectation-maximization can be used to fill in the "missing" state paths.

2.3 Two Alternate Priors for CRFs

In previous unpublished experiments on named entity recognition, the first author found that obtaining high accuracy required that some features have large-magnitude λ weights, but that the Gaussian prior (which "punishes" weights by their value squared), was preventing them from reaching this magnitude. At the same time, the Gaussian prior was only very weakly "punishing" low-magnitude weights, and thus the resulting models had many non-zero weights, but almost none above one.

Since the number of features in CRFs is often created by a large power set of conjunctions, intuitively one would expect that many features are irrelevant, and should have zero weight, while others that have sufficient training-data support should be allowed more freedom in their range of weight value.

A linear (L1, abs(x)-shaped), rather than quadratic (L2, x^2-shaped) penalty function would come closer to this goal. It is well-known, for example, in the support-vector machine literature, that L1 loss functions encourage "sparse" solutions, in which many weights for unimportant features are nearly zero.

The absolute value function is not everywhere differentiable, being undefined at zero. However, the function $(a/b)\log(\cosh(b\,\lambda))$ is everywhere differentiable, and has an appealingly similar shape, as well as two intuitively meaningful meta-parameters: a controls the slope of the line away from zero, and b controls the "roundness" of the differentiable curve close to zero. The derivative of this function is $a\,b\tanh(\lambda)$. We call this a *hyperbolic-L1* prior; it has likely been used elsewhere on other contexts, but we have not yet found reference to it.

An even more extreme embodiment of "sparseness" and non-interference with other weights would be a weight penalty function that rises away from zero as previously, but becomes more flat further out. This shape might be captured with a prior such as $(a/b)\log(\cosh(b\,\lambda))/\mathrm{abs}(\lambda/c)$, where c is a parameter that controls how far away from zero the flattening begins.

We have experimented with the first, but not yet the second. As will be described in section 5, on Chinese word segmentation, we did not find the hyperbolic-L1 prior to improve performance.

3 Use of Domain Knowledge with CRFs

One of the chief strengths of CRFs is their ability to easily incorporate arbitrary features of the input sequence. This strength can be used with traditional HMM sequence modeling features, *i.e.* the current token identity,[1] by simply adding additional features that query the identity of tokens before and after the current token, as well as conjunctions of these features. These additional features are highly non-independent and would be quite problematic for a generative model like an HMM—not to mention the outright impossibility of a generative sequence model depending on *future, not yet generated* elements of the sequence.

However, these type of features still do not take full advantage of CRFs' flexibility. This paper strongly advocates learning that is not *tabula rasa*, but rather incorporates as much domain knowledge as possible. Doing so can improve generalization and also greatly reduce the amount of labeled training data that is necessary. In language technology applications, which often use models with extremely large numbers of parameters, the need for large volumes of labeled training data is often a bottleneck to high accuracy. Methods that allow the straightforward incorporation of domain knowledge provide an extra avenue for weak supervision, and reduce the burden of data labeling.

Domain knowledge comes in many forms, nearly all of which can be cast as features: gazeteers and lexicons, lists of significant token suffixes and other sub-constituents, hand-generated questions about entire regions of the sequence, even the output of previous hand-built rules, expert systems and ad-hoc solutions.

[1] for example, a feature defined to be non-zero only when the current token is the Chinese character for "cat")

Input: (1)Training set: sentences of Chinese text with hand-segmented words, and (2) several lexicons of Chinese words and characters belonging to different salient categrories (such as location names, surnames, digits, etc).
Algorithm: Use a quasi-Newton method to set the parameters of a CRF finite state model to maximize the conditional likelihood of START and NONSTART tags on the characters in the training data.
Output: A finite state model that finds word boundaries in unsegmented Chinese text by finding the most likely state (and thus, tag) sequence using the Viterbi algorithm.

Figure 1: Outline of our algorithm.

The next two sections describe such an approach to Chinese word segmentation, and show positive experimental results, including high accuracies with extremely small amounts of labeled data.

4 CRFs plus Domain Knowledge for Chinese Word Segmentation

Our approach to Chinese word segmentation combines the strengths of (1) supervised machine learning trained on a hand-segmented corpus, (2) a rich variety of domain-knowledge-laden lexicons which reduce the need for large quantities of labeled data, (3) a maximum-likelihood training method gauranteed to find the global optima, (4) a strong representation of contextal information to help correctly handle out-of-vocabulary (OOV) words, (5) and the full statistical power of finite-state inference.

We cast the segmentation problem as one of sequence tagging. Chinese characters that begin a new word are given the START tag, and characters in the middle and at the end of words are given the NONSTART tag. The task of segmenting new, unsegmented test data becomes a matter of assigning to it a sequence of tags (labels).

A conditional random field are configured as a finite state machine (FSM) for this purpose, and tagging is performed using Viterbi to find the most likely label sequence for a given character sequence. The FSM may be first-order Markov, in which case there is one state for each label, and the choice of next label depends only on the single previous label and the input sequence. The FSM may also

be second-order Markov or higher. A second-order FSM has a state for every possible pair of labels, and the choice of next label depends on the previous two labels.

Some previous methods have used a single large lexicon of Chinese words, or a small collection of "character class" definitions (such as digits and punctuation marks) to help perform segmentation, *e.g.* (Ponte and Croft, 1996; Xue and Converse, 2002). CRFs allow us to take this approach to the extreme: we incorporate not a few, but a large variety of different lexicons, each with different specialities and meanings. For example, one is simply a very large dictionary of Chinese words, another is a list of location names, others capture surnames, digits, etc. The complete list our lexicons is given in the next section.

The features used in our CRF are membership queries of character subsegments in these lexicons; for example, one feature might indicate that the subsequence consisting of the previous character and the current character appears in the lexicon of locations. We also add features that query lexicon membership of subsequences before and after the current character. For example, a feature could ask if the character three positions ahead appears in the lexicon of organization name indicators. Furthermore we create new features out of two- and three-element conjunctions of these atomic features, that might ask, for example, if the character two positions previous is in the list of prepositions, and is the character one position ahead in the list of verbs.

Because features that examine past and future provide strong contextual clues, and because we use the power of Viterbi for finite state inference, we can usually successfully segment new words that don't appear in any of our lexicons. All our features are binary-valued. Although we could also add conjuncts indicating the lack of membership in a lexicon, we have not yet done this.

5 Experimental Results

We tested our approach using hand-segmented data from the Penn Chinese Treebank (LDC-2001T11). It consists of 325 Xinhua newswire articles on a variety of subjects. We selected it because it is a standard data set on which others have published results, and because it has long sentences, relatively few numbers and large and rich vocabulary—all of which make word segmentation more challenging.

In pre-processing this Chinese Treebank data, we ignore all part-of-speech tags, and exclude all header and title information, obtaining our training and testing data from only the body of the articles. The article bodies contain a total of 3,811 sentences, comprising 96,653 words (10,653 of which are unique), and 166,120 Chinese characters.

The test set for all experiments below consists of the sentences of the articles whose file number in the Treebank is evenly divisible by four. The sentences from the other articles comprise the training set. The training set contains 2,805 sentences (71,969 tokens[2], 62,145 words); the testing set, 1,006 sentences (24,684 tokens, 21,109 words). The testing set has 4,587 unique words, 1,688 (37%) of which do not appear anywhere in the training set. This out-of-vocabulary rate speaks to the large and varied vocabulary in this data set, and the difficulty of this segmentation task.

5.1 Lexicon Features as Domain Knowledge

Many lexicons of Chinese words and characters are readily available from the Internet and other sources. Our domain-knowledge-providing lexicons consist of 28 lists of Chinese words and characters obtained from several Internet sites,[3] from BBN's Chinese named entity recognizer, and from a local native speaker. They were all obtained independently of LDC data. (We could have reasonably used the Penn Chinese Dictionary and a word lexicon built from the training corpus, but did not). The complete list of our lexicons appears in figure 2.

Features used in our CRF are time-shifted conjunctions of membership queries to these lexicons of character subsequence windows size 1-8. An additional set of features captures whether the current character is equal to the character before or after it, using shifts of up to 3 positions. Using the notation g_t to indicate an individual feature, g, tested at sequence position t, and $g_t h_{t'} i_{t''}$ indicate a conjunction of three feature tests applied at various sequence positions, the complete set of time-shifting patterns we use is: g_t, g_{t+1}, g_{t-1}, g_{t+2}, g_{t-2}, g_{t+3},

[2] words plus punctuation

[3] *e.g.* http://www.mandarintools.com.

adjective ending character	money
adverb ending character	negative characters
building words	organization indicator
Chinese number characters	preposition characters
Chinese period (dot)	provinces
cities & regions	punctuation characters
countries	Roman alphabetics
dates	Roman digits
department characters	stopwords
digit characters	surnames
foreign name characters	symbol characters
function words	time
job titles	verb characters
locations	wordlist (188k lexicon)

Figure 2: Lexicons used in our experiments

Method	# training sentences	training tag acc.	testing segmentation prec.	recall	F1
Peng	$\sim$ 5M	?	75.1	74.0	74.2
Ponte	?	?	84.4	87.8	86.0
Teahan	$\sim$40k	?	?	?	94.4
Xue	$\sim$10k	97.8	95.2	95.1	95.2
CRF	2805	99.9	97.3	97.8	**97.5**
CRF	1402	99.9	96.9	97.4	97.1
CRF	140	100	95.4	96.0	95.7
CRF	56	100	93.9	95.0	94.4
CRF	7	100	82.8	83.5	83.1
CRF+	2805	100	99.6	99.7	99.6
CRF-	2805	99.9	92.6	92.6	92.6
CRF-	1402	100	90.6	90.4	90.5
CRF-	140	100	80.0	78.6	79.3
CRF-	7	100	60.4	59.0	59.7

Figure 3: Accuracy comparison in percentages. *Peng* (Peng and Schuurmans, 2001), is an unsupervised method. *Ponte* (Ponte and Croft, 1996), is a lexicon-based method. *Teahan* (Teahan et al., 2000), is a supervised generative FSM, *Xue* (Xue and Converse, 2002), is a supervised maximum-entropy method. *CRF* is the method described in this paper. *CRF+* is the same, using an artificially complete lexicon that has a zero OOV rate. *CRF-* uses traditional character-based features instead of multiple lexicons.

g_{t-3}, $g_t h_t$, $g_t h_{t+1}$, $g_{t-1} h_t$, $g_t h_{t+2}$, $g_{t-2} h_t$, $g_t h_{t+3}$, $g_{t-3} h_t$, $g_{t-1} h_t i_{t+1}$. The total number of such features appearing in the training set is 42,187. Given that each of these features has a parameter on each FSM transition, and that there are sequence-start and -end weights for each state, our second-order Markov CRF has a total of 337,500 parameters.

Using our Java implementation and a 1.6MHz Pentium, training the CRF by L-BFGS on the 2805 sentences of the training set takes about 100 iterations, 300 megabytes of memory and 5 hours. Segmenting one sentence from the test set takes less than a tenth of a second.

Accuracy is measured in terms of percentage tags correct, and segmentation precision, recall and F1. Let P be the number of predicted segments, T be the number of true segments, and C be the number of predicted segments with perfect starting and ending boundaries; then precision is C/P, recall is C/T. F1 is the harmonic mean of precision and recall.

Results are given in Figure 3. On Chinese Treebank with realistic, incomplete lexicons, the best previous result of which we are aware is 95.2% F1 (Xue and Converse, 2002). CRFs produce 97.5% F1, reducing error by half. Furthermore, this improved performance was reached with significantly less training data. As additional testament to the training-data efficiency of our domain-knowledge-laden features, notice that CRFs maintain high accuracy with a shockingly small amount of training data: with only 280 training sentences, CRFs still out-perform previous methods; with only 56 labeled sentences, they match the performance of Teahan *et al.* (2000), which was trained on roughly 700-times as much data.

We were particularly pleased to see that CRF accuracy is high even on words that appear in none of our 30 lexicons. On the test set, there are 1134 out-of-vocabulary words, 985 of which the CRF correctly segments—a success rate of 87%. Brief scanning of our results by a native speaker indicates that some of the "errors" made by the CRF are actually judgement calls that could go either way, with most of the true segmentation errors being in proper names.

With the full training set CRFs are still over-fitting (F1 on the training set is 99.7%), so CRFs would be expected to do even better with more training data. We might also do significantly better if we gathered more words for our wordlist lexicon—when we artificially give CRFs a perfect lexicon by including all words in our training and testing sets (CRF+ in Figure 3), we obtain 99.6% F1.

To further demonstrate the advantages of domain-knowledge-laden features, we also trained and tested CRFs using only traditional character-based features

(CRF- in Figure 3). Features indicate the presence of individual characters, as well as time-shifted conjunctions, as before, with the following conjunction patterns: g_t, g_{t+1}, g_{t-1}, g_{t+2}, g_{t-2}, $g_t h_{t+1}$, $g_{t-1} h_t$—resulting in about 400k features. Including more context would have greatly enlarged the total number of features, which wouldn't appear helpful, since the model was already badly over-fitting, (having training F1 of 99.9% even with the maximum amount of training data).

All of the above results use a Gaussian prior with variance $\sigma = 3$, a value selected by cross-validation on the training set. Alternative values of σ running from 0.1 to 10 did not have a high impact—all had testing F1 above or near 97%. Experiments with the hyperbolic-L1 prior also yielded results in the same range, with none higher than 97.3%.

6 Conclusions

Conditional random fields support a strong, principled integration of finite-state inference, discriminative training, and the use of rich, arbitrary features. When domain knowledge is incorporated into the feature set, we can drastically reduce the need for labeled training data.

We would expect our accuracy to rise even further by performing cross-validated error analysis on the training set, and by tuning the features and other meta-parameters. It should also improve by further enlarging our lexicons and increasing our amount of labeled training data. In future work we also plan to investigate the benefits of feature induction (which could be used to discover those cases when capturing individual Chinese characters would be helpful), and investigate methods for training with combinations of labeled and unlabeled data. Additionally, we plan to apply our methods to segmenting other Asian languages, such as Japanese and Thai.

Acknowledgements

This work was supported in part by the Center for Intelligent Information Retrieval and in part by SPAWARSYSCEN-SD grant numbers N66001-99-1-8912 and N66001-02-1-8903. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsor. We would also like to thank Douglas Waterman for helpful comments on a previous draft.

References

Rie Kubota Ando and Lillian Lee. 2000. Mostly-unsupervised statistical segmentation of Japanese: Applications to Kanji. *NAACL*, pages 241–248.

R. H. Byrd, J. Nocedal, and R. B. Schnabel. 1994. Representations of quasi-Newton matrices and their use in limited memory methods. *Mathematical Programming*, 63:129–156.

K.J. Chen and S.H. Liu. 1992. Word identification for Mandarin Chinese sentence. In *Proceedings of the Fourteenth International Conference on Computaional Linguistics*.

Stanley F. Chen and Ronald Rosenfeld. 1999. A Gaussian prior for smoothing maximum entropy models. Technical Report CMU-CS-99-108, CMU.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. ICML*, pages 282–289.

Robert Malouf. 2002. A comparison of algorithms for maximum entropy parameter estimation.

Andrew McCallum, Dayne Freitag, and Fernando Pereira. 2000. Maximum entropy Markov models for information extraction and segmentation. In *Proc. ICML 2000*, pages 591–598.

Fuchun Peng and Dale Schuurmans. 2001. Self-supervised Chinese word segmentation. In F. Hoffman et al., editor, *Advances in Intelligent Data Analysis*, pages 238–247.

Jay M. Ponte and W. Bruce Croft. 1996. Useg: A retargetable word segmentation procedure for information retrieval. *Symposium on Document Analysis and Information Retrieval*.

Lawrence R. Rabiner. 1990. A tutorial on hidden markov models and selected applications in speech recognition. In Alex Weibel and Kay-Fu Lee, editors, *Readings in Speech Recognition*, pages 267–296. Morgan Kaufmann.

Fei Sha and Fernando Pereira. 2002. Shallow parsing with conditional random fields. Technical Report CIS TR MS-CIS-02-35, University of Pennsylvania, Computer and Information Science Department. (forthcoming).

Richard W. Sproat, Chilin Shih, William Gale, and Nancy Change. 1996. A stochastic finite-state word-segmentation algorithm for chinese. *Computational Linguistics*, 22(3):377–404.

W. J. Teahan, Y. Wen, R. McNab, and I. H. Witten. 2000. A compression-based algorithm for Chinese word segmentation. *Computational Linguistics*, 26(3):375–393.

Zimin Wu and Gwyneth Tseng. 1993. Chinese text segmentation for text retrieval: Achievements and problems. *Journal of the American Society for Information Science*, 44(9):532–544.

Nianwen Xue and Susan Converse. 2002. Combining classifiers for Chinese word segmentation. In *Proceedings of the SIGHAN Workshop*.