Chinese Word Segmentation and Named Entity Recognition: A Pragmatic Approach

Jianfeng Gao*
Microsoft Research Asia

Mu Li*
Microsoft Research Asia

Andi Wu+
GrapeCity Inc.

Chang-Ning Huang*
Microsoft Research Asia

This article presents a pragmatic approach to Chinese word segmentation. It differs from most previous approaches mainly in three respects. First, while theoretical linguists have defined Chinese words using various linguistic criteria, Chinese words in this study are defined pragmatically as segmentation units whose definition depends on how they are used and processed in realistic computer applications. Second, we propose a pragmatic mathematical framework in which segmenting known words and detecting unknown words of different types (i.e., morphologically derived words, factoids, named entities, and other unlisted words) can be performed simultaneously in a unified way. These tasks are usually conducted separately in other systems. Finally, we do not assume the existence of a universal word segmentation standard that is application-independent. Instead, we argue for the necessity of multiple segmentation standards due to the pragmatic fact that different natural language processing applications might require different granularities of Chinese words.

These pragmatic approaches have been implemented in an adaptive Chinese word segmenter, called ***MSRSeg,*** *which will be described in detail. It consists of two components: (1) a generic segmenter that is based on the framework of linear mixture models and provides a unified approach to the five fundamental features of word-level Chinese language processing: lexicon word processing, morphological analysis, factoid detection, named entity recognition, and new word identification; and (2) a set of output adaptors for adapting the output of (1) to different application-specific standards. Evaluation on five test sets with different standards shows that the adaptive system achieves state-of-the-art performance on all the test sets.*

1. Introduction

This article is intended to address, with a unified and *pragmatic* approach, two fundamental questions in Chinese natural language processing (NLP): What is a 'word' in Chinese?, and How does a computer identify Chinese words automatically? Our approach is distinguished from most previous approaches by the following three unique

* Natural Language Computing Group, Microsoft Research Asia, 5F, Sigma Center, No. 49, Zhichun Road, Beijing, 100080, China. E-mail: jfgao@microsoft.com, muli@microsoft.com, cnhuang@msrchina.research.microsoft.com.

+ The work reported in this article was done while the author was at Microsoft Research. His current e-mail address is andi.wu@grapecity.com.

Submission received: 22 November 2004; revised submission received: 20 April 2005; accepted for publication: 17 June 2005.

components that are integrated into a single model: a taxonomy of Chinese words, a unified approach to word breaking and unknown word detection, and a customizable display of word segmentation.[1] We will describe each of these in turn.

Chinese word segmentation is challenging because it is often difficult to define what constitutes a word in Chinese. Theoretical linguists have tried to define Chinese words using various linguistic criteria (e.g., Packard 2000). While each of those criteria provides valuable insights into "word-hood" in Chinese, they do not consistently lead us to the same conclusions. Fortunately, this may not be a serious issue in computational linguistics, where the definition of words can vary and can depend to a large degree upon how one uses and processes these words in computer applications (Sproat and Shih 2002).

In this article, we define the concept of Chinese words from the viewpoint of computational linguistics. We develop a taxonomy in which Chinese words can be categorized into one of the following five types: lexicon words, morphologically derived words, factoids, named entities, and new words.[2] These five types of words have different computational properties and are processed in different ways in our system, as will be described in detail in Section 3. Two of these five types, factoids and named entities, are not important to theoretical linguists but are significant in NLP.

Chinese word segmentation involves mainly two research issues: word boundary disambiguation and unknown word identification. In most of the current systems, these are considered to be two separate tasks and are dealt with using different components in a cascaded or consecutive manner.

However, we believe that these two issues are not separate in nature and are better approached simultaneously. In this article, we present a unified approach to the five fundamental features of word-level Chinese NLP (corresponding to the five types of words described earlier): (1) word breaking, (2) morphological analysis, (3) factoid detection, (4) named entity recognition (NER), and (5) new word identification (NWI). This approach is based on a mathematical framework of linear mixture models in which component models are inspired by the source–channel models of Chinese sentence generation. There are basically two types of component models: a source model and a set of channel models. The source model is used to estimate the generative probability of a word sequence in which each word belongs to one word type. For each of the word types, a channel model is used to estimate the likelihood of a character string, given the word type. We shall show that this framework is flexible enough to incorporate a wide variety of linguistic knowledge and statistical models in a unified way.

In computer applications, we are more concerned with *segmentation units* than *words*. While words are supposed to be unambiguous and static linguistic entities, segmentation units are expected to vary from application to application. In fact, different Chinese NLP-enabled applications may have different requirements that request different granularities of word segmentation. For example, automatic speech recognition (ASR) systems prefer longer "words" to achieve higher accuracy, whereas in-

1 In this article, we differentiate the terms *word breaking* and *word segmentation*. Word breaking refers to the process of segmenting known words that are predefined in a lexicon. Word segmentation refers to the process of both lexicon word segmentation and unknown word detection.

2 New words in this article refer to out-of-vocabulary words that are neither recognized as named entities or factoids nor derived by morphological rules. These words are mostly domain-specific and/or time-sensitive (see Section 5.5 for details).

formation retrieval (IR) systems prefer shorter "words" to obtain higher recall rates (Wu 2003).

Therefore, we do not assume that an application-independent universal word segmentation standard exists. We argue instead for the existence of multiple segmentation standards, each for a specific application. It is undesirable to develop a set of application-specific segmenters. A better solution would be to develop a generic segmenter with customizable output that is able to provide alternative segmentation units according to the specification that is either predefined or implied in the application data. To achieve this, we present a transformation-based learning (TBL; Brill 1995) method, to be described in Section 6.

We implement the pragmatic approach to Chinese word segmentation in an adaptive Chinese word segmenter called **MSRSeg.** It consists of two components: (1) a generic segmenter that is based on the linear mixture model framework of word breaking and unknown word detection and that can adapt to domain-specific vocabularies, and (2) a set of output adaptors for adapting the output of (1) to different application-specific standards. Evaluation on five test sets with different standards shows that the adaptive system achieves state-of-the-art performance on all the test sets. It thus demonstrates the possibility of a single adaptive Chinese word segmenter that is capable of supporting multiple applications.

The remainder of this article is organized as follows. Section 2 presents previous work in this field. Section 3 introduces the taxonomy of Chinese words and describes the corpora we used in our study. Section 4 presents some of the theoretical background on which our unified approach is based. Section 5 outlines the general architecture of the Chinese word segmenter, MSRSeg, and describes each of the components in detail, presenting a separate evaluation of each component where appropriate. Section 6 presents the TBL method of standards adaptation. While in Section 5 we presume the existence of an annotated training corpus, we focus in Section 7 on the methods of creating training data in a (semi-)automatic manner, with minimal or no human annotation. We thus demonstrate the possibilities of unsupervised learning of Chinese words. Section 8 presents several evaluations of the system on the different corpora, each corresponding to a different segmentation standard, in comparison with other state-of-the-art systems. Finally, we conclude the article in Section 9.

2. Previous Work

2.1 Approaches to Word Segmentation

Many methods of Chinese word segmentation have been proposed: reviews include Wu and Tseng (1993); Sproat and Shih (2002); and Sun and Tsou (2001). These methods can be roughly classified as either dictionary-based or statistically-based methods, while many state-of-the-art systems use hybrid approaches.

In dictionary-based methods, given an input character string, only words that are stored in the dictionary can be identified. One of the most popular methods is maximum matching (MM), usually augmented with heuristics to deal with ambiguities in segmentation. Studies that use this method or minor variants include Chen et al. (1999) and Nie, Jin, and Hannan (1994). The performance of these methods thus depends to a large degree upon the coverage of the dictionary, which unfortunately may never be complete because new words appear constantly. Therefore, in addition to the dictionary, many systems also contain special components for unknown word identification.

In particular, statistical methods have been widely applied because they use a probabilistic or cost-based scoring mechanism rather than a dictionary to segment the text. These methods have three drawbacks. First, some of these methods (e.g., Lin et al. 1993; Chang and Su 1997) identify OOV (out-of-vocabulary) words without identifying their types. For instance, one might identify a string as a unit but fail to identify that it is a person name. Second, many current statistical methods do not incorporate linguistic knowledge effectively into segmentation. For example, Teahan et al. (2000) and Dai et al. (1999) do not use any linguistic knowledge. Thus, the identified OOV words are likely to be linguistically implausible, and consequently, additional manual checking is needed for some subsequent tasks such as parsing. Third, in many current segmenters, OOV identification is considered a separate process from segmentation (e.g., Chen 2003; Wu and Jiang 2000; Chen and Bai 1998). For instance, Chen (2003) assumes that OOV words are usually two or more characters long and are often segmented into single characters. He then uses different components to detect OOV words of different types in a cascaded manner after the basic word segmentation.

We believe that the identification of OOV words should not be treated as a problem separate from word segmentation. We propose a unified approach that solves both problems simultaneously. A previous work along this line is Sproat et al. (1996), which is based on weighted finite-state transducers (FSTs). Our approach is similarly motivated but is based on a different mechanism: linear mixture models. As we shall see, the models provide a more flexible framework for incorporating various kinds of lexical and statistical information. Many types of OOV words that are not covered in Sproat's system can be dealt with in our system. The linear models we used are originally derived from linear discriminant functions widely used for pattern classification (Duda, Hart, and Stork 2001) and have been recently introduced into NLP tasks by Collins and Duffy (2001). Other frameworks of Chinese word segmentation, which are similar to the linear models, include maximum entropy models (Xue 2003) and conditional random fields (Peng, Feng, and McCallum 2004). They also use a unified approach to word breaking and OOV identification.

2.2 More on New Word Identification

In this article, we use the term "new words" to refer to OOV words other than named entities, factoids, and morphologically derived words. "New words" are mostly domain-specific terms (e.g., 蜂窝式 'cellular') and time-sensitive political, social, or cultural terms (e.g., 三通 'Three Links'; 非典 'SARS'). There have been two general approaches to NWI. The first is to acquire new words from large corpora off-line and put them into a dictionary before word segmentation starts (e.g., Fung and Wu 1994; Nie, Jin and Hannan 1994; Chien 1997; Gao et al. 2002). The other is to detect new words on-line, i.e., to spot new words in a sentence on the fly during the process of word segmentation (e.g., Chen 2003; Wu and Jiang 2000). These two approaches complement one another, and we use both of them in our system.

There have been quite a few methods proposed for the off-line approach in which the basic assumption is that a Chinese word should appear as a stable sequence in the corpus. These methods use metrics that are based on statistical features such as mutual information, term frequency, and their variants. They require a reasonably large training corpus. The new words detected are mostly proper nouns and other relatively frequent words. Unfortunately, new words, under our definition of the term, may not be detected.

Fewer methods have been proposed for an on-line approach, and that is the focus of this article. Some recent advances in on-line NWI explore the use of machine learning approaches. For example, Li et al. (2003) define NWI as a binary classification problem and use support vector machines (SVM) to combine various linguistically motivated features to determine whether a Chinese character sequence is a word. Our method is an extension of that of Li et al. in that NWI is not a stand-alone process in our system but an integral part of word segmentation. We shall show experimentally the benefit of the integration in Section 5.5.

2.3 Standards Adaptation

As described earlier, while Chinese words are supposed to be well-defined, unambiguous, and static linguistic entities, we are more concerned with *segmentation units* that are expected to vary among different computer applications. This inspires the development of an adaptive Chinese word segmenter.

However, most of the previous segmenters have been developed according to a standard that assumes a single *correct* segmentation. The only adaptive system, to the best of our knowledge, is the customizable segmenter described in Wu (2003), in which the display of the segmentation output can be customized by users.[3] The adaptation method we will describe in Section 6 can be viewed as an improved version in that the adaptation rules (or transformations) are acquired automatically from application data via the TBL method (Gao et al. 2004). Though the use of TBL for Chinese word segmentation is not new (see Palmer [1997]; Hockenmaier and Brew [1998]), none of the previous work is aimed at standards adaptation.

2.4 Evaluation

The performance of Chinese word segmenters is generally reported in terms of precision and recall. However, a comparison across systems could be very difficult for two reasons. First, the "correct" segmentation is not clearly defined. It is common that for a given sentence there are multiple plausible word segmentations. As shown in Sproat et al. (1996), the rate of agreement between two human judges is less than 80%. To deal with this problem, Fung and Wu (1994) suggest a procedure called *nk-blind* that uses n blind judges' standards. If we set $k = 1$, it is sufficient for a segmentation to be considered correct if it agrees with at least one of the n judges. If $k = n$, all judges must agree. Therefore, *nk-blind* gives a more representative performance measure by taking into account multiple judges. Similarly, Sproat et al. (1996) also uses multiple human judges. In Section 8.2, we will present our method for cross-system comparison. We do not use multiple human judges. Instead, we only consider a set of measures that are lexicon-independent and less ambiguous among different human judges and systems.

The second reason that cross-system comparisons are difficult concerns the use of different test sets and ground rules by many research papers. For example, some papers report precision and recall rates of 98% or 99%. But they either count only the words that are stored in the dictionary or use unrealistically simple data with a very low OOV rate. Recently, the ACL-SIGHAN–sponsored First International Chinese Word Segmentation

3 Huang et al. (1997) also proposed an adaptive, three-level standard.

Bakeoff alleviated the situation to some degree (Sproat and Emerson 2003). The Bakeoff released four data sets, each corresponding to a different standard, and consistent train–test splits. We evaluate our segmenter using those four data sets in Sections 6.2 and 8.3.

3. Chinese Words

This section defines Chinese words at three levels. We begin with a taxonomy by which Chinese words are categorized into five main types according to the way they are processed and used in realistic systems. Second, we develop the MSR standard, which is a set of specific rules to guide human annotators in segmenting Chinese sentences. Finally, we describe the development of a gold test set and how we evaluate Chinese word segmenters. Here, we use the term "gold test set" to refer to the manually annotated corpus, according to the MSR standard, on top of the "test corpus" that is the raw text corpus.

3.1 Taxonomy

The taxonomy of Chinese words is summarized in Table 1, where Chinese words are categorized into the five types: entries in a lexicon (or lexicon words, LWs), morphologically derived words (MDWs), factoids (FTs), named entities (NEs), and new words

Table 1
Taxonomy of Chinese words used in developing MSRSeg.

LW	Lexical Word	教授, 朋友, 高兴, 吃饭
MDW (see Figure 1 for details)	Morphologically Derived Word	
MP_*, MS_*	Affixation (Prefix, Suffix)	朋友们
MR_*	Reduplication	高高兴兴
ML_*	Splitting	吃了饭
MM_*	Merging	上下班
MHP_*	Head + Particle	走出去
FT	Factoid word	
Dat	Date	10 月11 日
Dur	Duration	20 多分钟
Tim	Time	十二点三十分
Per	Percent and fraction	60%, 1/8
Mon	Money	25000 美元
NUMBER	Frequency, integer, decimal, ordinal, rate, etc.	三次, 三个, 12.2 亿, 第二个
MEASURE	Age, weight, length, area, capacity, speed, temperature, angle, etc.	78 岁, 700 公斤, 2 亿公里, 1.8 亿公顷, 78 亿立方米, 每秒 2.89 米, 9 摄氏度, 5 度
Ema	E-mail	jfgao@microsoft.com
Pho	Phone, fax, telex	62617711
www	WWW	www.microsoft.com
NE	Named Entity	
P	Person name	李俊生, 亚历山大
L	Location name	蒙特利尔, 圣海伦岛公园
O	Organization name	中央民族乐团, 毕加索博物馆
NW	New Word	三通, 非典

(NWs). These five types of words have different functions in Chinese NLP and are processed in different ways in our system. For example, a plausible word segmentation for the sentence in Figure 1(a) is shown in the same figure. Figure 1(b) is the output of our system, where words of different types are processed in different ways:

- For LWs, word boundaries are detected, such as 教授 'professor'.
- For MDWs, their morphological patterns and stems are detected, e.g., 朋友们 'friend+s' is derived by affixation of the plural affix 们 to the noun (stem) 朋友 (MA_S indicates a suffixation pattern), and 高高兴兴 'happily' is a reduplication of the stem 高兴 'happy' (MR_AABB indicates an AABB reduplication pattern).
- For FTs, their types and normalized forms are detected, e.g., 12:30 is the normalized form of the time expression 十二点三十分 ("tim" indicates a time expression).
- For NEs, their types are detected, e.g., 李俊生 'Li Junsheng' is a person name.

The five types of words cannot be defined by any consistent classification criteria (e.g., the relation between MDWs and LWs depends on the lexicon being used); the taxonomy therefore does not give a clear definition of Chinese words. We do not intend for this article to give a standard definition of Chinese words. Instead, we treat Chinese word segmentation as a preprocessing step where the best *segmentation units* depend on how they are used in the consuming applications. The five word types represent the five types of Chinese words that appear in most applications. This is one of the reasons that we title this article "a pragmatic approach." We focus on two tasks in the approach: processing of the five types of Chinese word using a unified framework that can be jointly optimized (Sections 4 and 5), and adapting our system to different applications (Section 6). Now, we describe each of the five word types in Table 1 in detail.

LWs (lexicon words): Although some previous research has suggested carrying out Chinese word segmentation without the use of dictionaries (e.g., Sproat and Shih 1990; Sun, Shen and Tsou 1998), we believe that a dictionary is an essential component of many applications. For example, in a machine translation system, it is desirable to segment a sentence into LWs as much as possible so that the candidate translations of these words can be looked up in a bilingual dictionary. Similarly, we would also like to segment a sentence into LWs in a Chinese text-to-speech (TTS) system because the pronunciations stored in the dictionary are usually much more precise than those generated dynamically (for instance, by character-to-sound rules). In our system, we used a lexicon containing 98,668 words, including 22,996 Chi-

(a) 朋友们/十二点三十分/高高兴兴/地/到/李俊生/教授/家/吃饭
(Friends happily go to professor Li Junsheng's home for lunch at twelve thirty.)

(b) [朋友+们 MA_S] [十二点三十分 12:30 tim] [高兴 MR_AABB] [地][到] [李俊生 P] [教授] [家] [吃饭]

Figure 1
(a) A Chinese sentence. Slashes indicate word boundaries. (b) An output of our word segmentation system. Square brackets indicate word boundaries. + indicates a morpheme boundary.

nese characters stored as single-character words. This lexicon is a combination of several dictionaries authored by Chinese linguists and used in different Microsoft applications. Thus, all LWs in theory are similar to those described in Packard (2000), i.e., linguistic units that are "salient and highly relevant to the operation of the language processor."

MDWs (morphologically derived words): Chinese words of this type have the following two characteristics. First, MDWs can be generated from one or more LWs (called stems) via a productive morphological process. For example, in Figure 2, the MDW 高高兴兴 'happily' is generated from a stem 高兴 'happy' via an AABB reduplication process. As shown in Table 1, there are five main categories of morphological processes, each of which has several subcategories, as detailed in Figure 2 (see Wu [2003] for a detailed description):

- **Affixation (MP** and **MS):** 朋友们 (friend - *plural*) 'friends';
- **Reduplication (MR):** 高兴 'happy' → 高高兴兴 'happily';
- **Splitting (MS)** (i.e., a set of expressions that are separate words at the syntactic level but single words at the semantic level): 吃了饭 'already ate', where the bi-character word 吃饭 'eat' is split by the particle 了 'already';
- **Merging (MM):** 上班 'on duty' + 下班 'off duty' → 上下班 'on-off duty'; and
- **Head Particle (MHP)** (i.e., expressions that are verb + comp): 走 'walk' + 出去 'out' → 走出去 'walk out'.

The second characteristic of MDWs is that they form stable Chinese character sequences in the corpus. That is, the components within the MDWs are strongly correlated (of high co-occurrence frequency), while the components at both ends have low correlations with words outside the sequence. We shall describe in Section 5.2 how the 'stability' of a Chinese sequence is measured qualitatively, and how to construct a morph-lexicon for Chinese morphology analysis.

FTs (factoids): There are ten categories of factoid words, such as time and date expressions, as shown in Table 1. All FTs can be represented as regular expressions. Therefore, the detection and normalization of FTs can be achieved by Finite State Machines.

NEs (named entities): NEs are frequently used Chinese names, including person names, location names, and organization names. One cannot develop a regular grammar that rejects or accepts the constructions of NEs with high accuracy, as we can do with most FTs. In Section 5.3, we shall describe how we use both heuristics and statistical models for NER.

NWs (new words): NWs are OOV words that are neither recognized as named entities or factoids nor derived by morphological rules. In particular, we focus on low-frequency new words, including newly coined words, occasional words, and mostly time-sensitive words (Wu and Jiang 2000). Many current segmenters simply ignore NWs, assuming that they are of little significance in most applications. However, we argue that the identification of those words is critical because a single unidentified word can cause segmentation errors in the surrounding words. For NLP applications that require full parsing, it is even more critical because a single error would cause a whole sentence to fail.

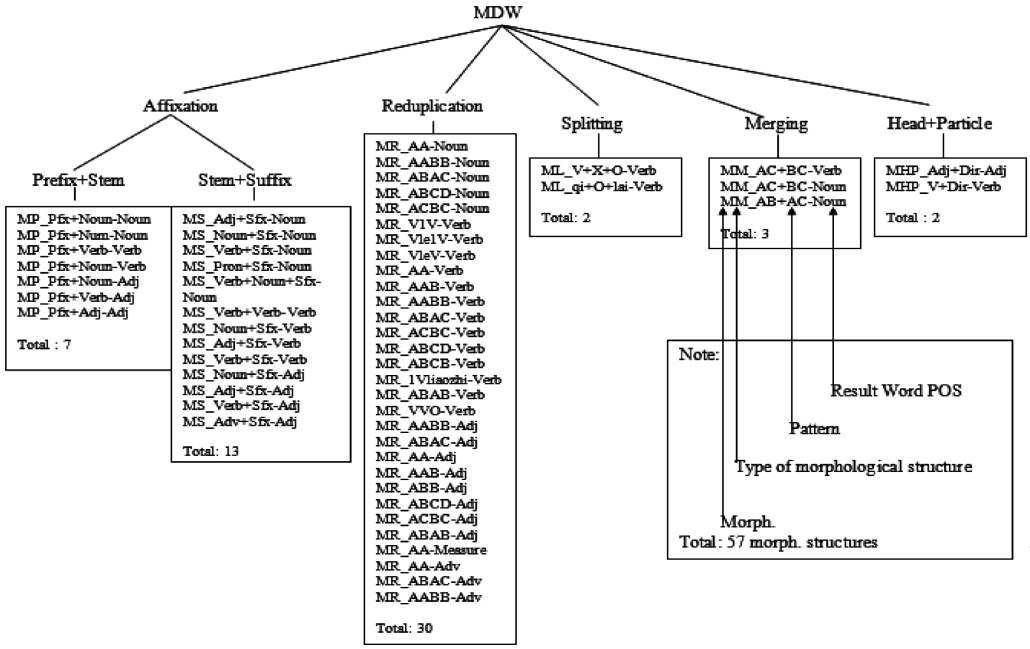


Figure 2
Taxonomy of morphologically derived words (MDWs) in MSRSeg.

3.2 MSR Standard

The taxonomy employed here has been specified in detail in the MSR standard. There are two general guidelines for the development of the standard:

1. The standard should be applicable to a wide variety of NLP tasks, of which some representative examples are Chinese text input, IR, TTS, ASR, and MT.
2. The standard should be compatible with existing standards, of which representative examples are the Chinese NE standards in ET/ER-99, the Mainland standard (GB/T), Taiwan's ROCLING standard (CNS14366; Huang et al. 1997), and the UPenn Chinese Treebank (Xia 1999), as much as possible.[4]

We are seeking a standard that is "linguistically felicitous, computationally feasible, and [ensures] data uniformity" (Huang et al. 1997; Sproat and Shih 2002). The MSR standard consists of a set of specific rules that aims at unambiguously determining the word segmentation of a Chinese sentence, given a reference lexicon. The development of the standard is an iterative procedure, interacting with the development of a gold test set (which we will describe in the next section). We begin with an initial set of

4 MET is a Chinese named entity standard defined in the MUC (message understanding conference, http://www.itl.nist.gov/iaui/894.02/related_projects/muc/). ER99 is an extension of MET, though it is not as universally used as MET (http://www.nist.gov/speech/tests/ie-er/er_99/er_99.htm).

segmentation rules, based on which four human annotators label a test corpus. Whenever an interannotator conflict is detected (automatically), we resolve it by revising the standard (e.g., mostly by adding more specific rules). The process is iterated until no conflict is detected. For example, we begin with the rule for detecting MDWs: "if a character sequence can be derived from a LW via a morphological process, then the sequence is treated as an MDW candidate." We then observe that both 吃了饭 'already ate' and 吃了一顿饭 'already had a meal' are derived from the LW 吃饭 'eat' via the morphological process of splitting. While 吃了饭 is a reasonable MDW, 吃了一顿饭 is debatable. We then add a rule: "MDW candidates with complex internal structures should be segmented." We also add a set of specific rules to define what a complex internal structure is. An example of those rules is "for MDW candidates of type MS, we only consider sequences that are less than four characters long."

One drawback of the approach is that the standard would become very complicated as we continue to add such specific rules, and people would start making clerical errors. We currently do not have a systematic solution to this. The complexity has to be controlled manually. That is, all new added rules are recompiled by a linguist so that the total number of rules is manageable.

3.3 MSR Gold Test Set and Training Set

Several questions had to be answered when we were developing the gold test set for evaluation.

1. How should we construct a test corpus for reliable evaluation?
2. Does the segmentation in the gold test set depend on a particular lexicon?
3. Should we assume a single correct segmentation for a sentence?
4. What are the evaluation criteria?
5. How should we perform a fair comparison across different systems using the gold test set?

We answer the first three questions in this section and leave the rest for Section 3.5. First, to conduct a reliable evaluation, we select a test corpus that contains approximately half a million Chinese characters that have been proofread and balanced in terms of domains and styles. The distributions are shown in Table 2. The gold test set is developed by annotating the test corpus according to the MSR standard via the iterative procedure described in Section 3.2. The statistics are shown in Table 3. Some fragments of the gold test set are shown in Figure 3, and the notation is presented in Table 1.

As discussed in Section 3.1, we believe that the lexicon is a critical component of many applications. The segmentation of the gold test set depends upon a reference lexicon, which is the combination of several lexicons that are used in Microsoft applications, including a Chinese text input system (Gao et al. 2002), ASR (Chang et al. 2001), TTS (Chu et al. 2003), and the MSR-NLP Chinese parser (Wu and Jiang 2000). The lexicon consists of 98,668 entries. We also developed a *morph-lexicon*, which contains 50,963 high-frequency MDWs. We will describe how the morph-lexicon was constructed in Section 5.2.

Regarding the third question, though it is common that there are multiple plausible segmentations for a given Chinese sentence, we keep only a single gold segmentation for each sentence for two reasons. The first is simplicity. The second is due to the fact

Table 2
Domain/style distribution in the MSR test corpus.

Domain/Style (MB)	Descriptive writing	Expository writing	Narrative writing	Practical writing[5]	Spoken	Total (%)
Culture		2.2	49.6	12.2		64 (6)
Economic		10.6	102.6	55.1	12.7	181 (17)
Literature	33.1	13.2	6.3			52.6 (5)
Military			42.1			42.1 (4)
Politics		36.2	102.9	88.8	100.8	328.7 (31)
Science & Tech.	7.3	14.7	85.8	2.1	9.4	119.3 (11)
Society	4.5	6.2	56.9	23.9		91.5 (8)
Sport		10.5	33.7		8.3	52.5 (5)
Computer		24.8	65.6			90.4 (9)
Law		2.1	26.3			28.4 (3)
Total (%)	44.9 (4)	120.5 (12)	571.8 (54)	182.1 (17)	131.2 (13)	1,051 (100)

Table 3
Words in the MSR gold test set.

Word type	N
LW	205,162
MDW	3,746
FT	6,630
NE:	
Person name	4,347
Location name	5,311
Organization name	3,850

that we currently do not know any effective way of using multiple segmentations in the above-mentioned applications. In particular, we segment each sentence as much as possible into words that are stored in the reference lexicon. When there are multiple segmentations for a sentence, we keep the one that contains the fewest number of words.

We have also manually developed a training set according to the MSR standard. It contains approximately 40 million Chinese characters from various domains of text such as newspapers, novels, and magazines. In our experiments, 90% of the training set is used for model parameter estimation, and the remaining 10% is a held-out set for tuning.

3.4 SIGHAN's Bakeoff Standards and Corpora

As mentioned in Section 1, MSRSeg is designed as an adaptive segmenter that consists of two components: (1) a generic segmenter that can adapt to different domain vocabularies, and (2) a set of output adaptors, learned from application data, for adapting to different application-specific standards.

5 Chinese writing is normally divided into the first three: descriptive, expository, and narrative. Practical writing is just an umbrella term for practical writing such as notes, letter, e-mails, and marriage announcements.

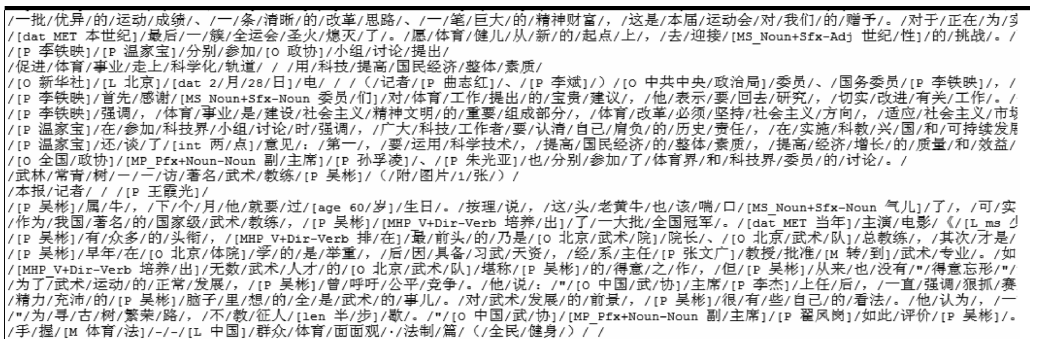


Figure 3
Fragments of the MSR gold test set.

Therefore, we evaluate MSRSeg using five corpora, each corresponding to a different standard, and consistent train–test splits, as shown in Table 4. MSR is described in previous sections, and the other four are standards used in SIGHAN's First International Chinese Word Segmentation Bakeoff (or Bakeoff for brevity) (Sproat and Emerson 2003). In the Bakeoff corpora, OOV is defined as the set of words in the test corpus not occurring in the training corpus.

In experiments, we always consider the following adaptation paradigm. Suppose we have a *general* predefined standard according to which we create a *large* amount of training data. We then develop a *generic* word segmenter. Whenever we deploy the segmenter for any application, we customize the output of the segmenter according to an application-specific standard that can be partially acquired from a given *small* amount of application data (called adaptation data).

The MSR standard described in Section 3.2 is used as the general standard in our experiments, on which the generic segmenter has been developed. The four Bakeoff standards are used as *specific* standards into which we wish to adapt the general standard. We notice in Table 4 that the adaptation data sets (i.e., training corpora for the four Bakeoff standards) are much smaller than the MSR training set. Thus, the experimental setting is a good simulation of the adaptation paradigm described above. In the rest of the article, we shall by default report results on the MSR data set unless otherwise stated.

3.5 Evaluation Methodology

As described earlier, we argue that Chinese words (or segmentation units) cannot be defined independently of the applications, and hence a more flexible system (i.e., an

Table 4
Standards and corpora.

Corpora (standards)	Abbrev.	# Training word	# Test word	OOV
MSR standard	**MSR**	20M	226K	.002
Beijing University	**PK**	1.1M	17K	.069
U. Penn Chinese Treebank	**CTB**	250K	40K	.181
Hong Kong City U.	**HK**	240K	35K	.071
Academia Sinica	**AS**	5.8M	12K	.021

adaptive segmenter such as MSRSeg) should be adopted. However, we are faced with the challenge of performing an objective and rigorous evaluation of such a system.

In general, the evaluation of NLP systems is concerned with both the criteria and the standard data sets. In this article, we argue that MSRSeg is a better system in two regards. First, the generic segmenter provides not only word segmentation but also word-internal structures (e.g., the tree structures of MDWs, FTs, and NEs, as will be described in Section 6) that cover all possible segmentations. Ideally, such a segmenter provides a *superset* of segmentation units where each different application can find the *subset* it needs. Second, the output adaptors of MSRSeg can automatically pick different subsets (i.e., segmentation units) from the superset according to different applications. Therefore, there are two criteria for evaluating an adaptive segmenter: how complete the superset is and how effective the adaptation is. The real evaluation will require some application data sets (i.e., segmented texts used by different applications). However, such application data are not available yet, and no other system has undergone such evaluation, so there is no way to compare our system against others in this fashion. The evaluation methodology we adopted in this article is a simulation. On the one hand, we developed a generic standard and a corresponding gold test set that simulates the generic superset that attempts to cover as many applications as possible. We then evaluate on the data set the completeness of the generic segmenter. On the other hand, we will show that we can effectively adapt the generic segmenter to the four different bakeoff data sets, each of which simulates an application subset.

The evaluation measures we use in this study are summarized in Table 5. The performance of MSRSeg is measured through multiple precision–recall (P/R) pairs, and F-measures (defined as 2PR/(P+R)), each for one word type. Riv is the recall of in-vocabulary words. Roov is the recall of OOV words. They are used to measure the segmenter's performance in resolving ambiguities in word segmentation and detecting unknown words, respectively. We also test the statistical significance of results, using the criterion proposed by Sproat and Emerson (2003).

In addition to Riv, the number of OAS (overlap ambiguity string) and CAS (combination ambiguity string) errors are used to measure the segmenter's performance of resolving ambiguities in word segmentation in more detail. Liang (1987) defines OAS and CAS as follows.

Definition 1
A character string ABC is called an **overlap ambiguity string** (OAS) if it can be segmented into two words either as AB/C or A/BC (not both), depending on context.

Table 5
Evaluation measures for Chinese word segmenter.

Measures	Remarks
P/R/F	Multiple pairs, each for one type of words (i.e., LW, MDW, FT, NE); P/R/F of NER are used for cross-system comparison
Roov	Test the performance of detecting unknown words
Riv	Test the performance of resolving ambiguities in word segmentation
# OAS errors	Similar to cross-bracketing, used for cross-system comparison
# CAS errors	Test on a set of 70 high-frequency CASs in our study
Significant test	See Sproat and Emerson (2003).

Definition 2
A character string AB is called a **combination ambiguity string** (CAS) if A, B, and AB are words.

Liang (1987) reports that the relative frequency of OASs is 1.2 per 100 characters in Chinese text, and the relative frequency of CASs is 12 times lower than that of OAS. However, according to the above definition, the relative frequency of CASs can be much higher because most single characters in Chinese can be words by themselves, and as a result, almost all two-character words can be CASs. However, this is not desirable. Consider the word 高度 'altitude'. Though both 高 'high' and 度 'degree' are words by themselves, the segmentation 高/度 almost never occurs in reality. To remedy this problem, Sun and Tsou (2001) revise the definition as follows:

Definition 3
A character string AB is called a **combination ambiguity string** (CAS) if A, B, and AB are words, and there is at least one context under which the segmentation A/B is plausible both semantically and syntactically.

Though the revision clarifies the definition in principle, it requires a judgment of the syntactic and semantic sense of the segmentation—a task where an agreement cannot be reached easily among different human annotators. Therefore, we only use the CAS measure in a pilot study. As will be described in Section 7, the number of CAS errors is estimated by counting the wrong segmentations of the predefined 70 high-frequency CASs.

While all the measures in Table 5 can be used in evaluating MSRSeg, most of them cannot be used in cross-system comparisons. For example, since the MSR gold test set is based on a reference lexicon, some of the measures are meaningless when we compare our system to other segmenters that use different lexicons. So in comparing different systems, we consider only the P/R/F of NER and the number of OAS errors (i.e., crossing brackets), because these measures are lexicon-independent and there is always a single unambiguous answer.

4. Theoretical Background

This section provides some theoretical background on the basis of the development of MSRSeg. We first present in Section 4.1 a Chinese word segmentation framework that uses source–channel models of Chinese sentence generation. Then, in Section 4.2, we generalize source–channel models as linear mixture models in which a wide variety of linguistic knowledge and statistical models can be incorporated in a unified way. These models are constructed via two basic modeling tools: (1) *n*-gram language models (LMs; Chen and Goodman 1999), and (2) finite state automata (FSA; Roche and Schabes 1997). More specifically, the LMs we used are bigram and trigram backoff models, where the parameters are estimated using maximum-likelihood estimation (MLE) with a particular smoothing method, called modified absolute discounting, described in Gao, Goodman, and Miao (2001). LMs are used to capture statistical information such as the likelihood of word or character sequence. FSAs are used to represent (1) the lexicon, (2) the rules for detecting FTs, and (3) the rules for generating NE candidates.

4.1 Source–Channel Models

The task of MSRSeg is to detect not only word boundaries but also word types so that words of different types can be processed as shown in Figure 1. Therefore, following the Chinese word taxonomy in Table 1, we define a Chinese *word class* as a group of words that are supposed to be generated according to the same distribution (or processed in the same manner) as follows:

1. Each LW is defined as a class;
2. Each MDW is defined as a class;
3. Each type of FT (e.g., time expressions) is defined as a class;
4. Each type of NE (e.g., person names) is defined as a class; and
5. All NWs belong to a class.

Notice that both the LW and MDW classes are open sets and that we need to assign a floor-value to those words that are not stored in the lexicons. In particular, we define six unknown word classes as follows. One class is used to represent all unknown LWs and all unknown MDWs whose type cannot be detected. The other five classes are used to represent unknown MDWs, one for each of the five types listed in Table 1, i.e., MP/MS, MR, MS, MM, and MHP. The probabilities of these unknown word classes are estimated using the Good-Turing method.

Let $\mathbf{w} = w_1 w_2 \ldots w_n$ be a word class sequence, and $\mathbf{s}$ be a Chinese sentence that is a character sequence. A segmenter's job is to choose the most likely word class sequence $\mathbf{w}^*$ among all possible candidates into which $\mathbf{s}$ could have been segmented:

$$\mathbf{w}^* = \arg\max_{\mathbf{w} \in \mathbf{GEN(s)}} P(\mathbf{w} \mid \mathbf{s}) = \arg\max_{\mathbf{w} \in \mathbf{GEN(s)}} P(\mathbf{w}) P(\mathbf{s} \mid \mathbf{w}) \tag{1}$$

where $\mathbf{GEN}(\mathbf{s})$ denotes the candidate set given $\mathbf{s}$.

Equation (1) is the basic form of source–channel models for Chinese word segmentation. The models assume that a Chinese sentence $\mathbf{s}$ is generated as follows: First, a person chooses a sequence of concepts (i.e., word classes $\mathbf{w}$) to be output, according to the probability distribution $P(\mathbf{w})$; then the person attempts to express each concept by choosing a sequence of characters, according to the probability distribution $P(\mathbf{s}|\mathbf{w})$.

The source–channel models also can be interpreted in another way: $P(\mathbf{w})$ is a stochastic model estimating the probability of word class sequence. It indicates, given a context, how likely a word class occurs. For example, person names are more likely to occur before a title such as 教授 'professor'. Consequently, we also refer to $P(\mathbf{w})$ as a *context model*. $P(\mathbf{s}|\mathbf{w})$ is a generative model estimating how likely a character string is generated given a word class. For example, the character string 李俊生 is more likely to be a person name than 里俊生 'Li Junsheng' because 李 is a common family name in China while 里 is not. So $P(\mathbf{s}|\mathbf{w})$ is also referred to as *class model*. In our system, we use only one context model (i.e., a trigram language model) and a set of class models of different types, each of which is for one class of words, as shown in Table 6.

Table 6
Context model, word classes, class models, and feature functions.

Word class	Model	Feature functions, $f(\mathbf{s}, \mathbf{w})$
Context model	Word class trigram, $P(\mathbf{w})$	$\log(P(\mathbf{w}))$
LW	Lexicon TRIE	Number of LW in $\mathbf{w}$.
MDW	Morph-lexicon TRIE	Number of MDW in $\mathbf{w}$.
NE	Character/word bigram, each for one type, $P(\mathbf{s}'\|\text{NE})$, where $\mathbf{s}'$ is a substring of $\mathbf{s}$, and forms an NE.	$\Sigma_{\mathbf{s}' \in \mathbf{A}} \log(P(\mathbf{s}' \mid \text{NE}))$, where $\mathbf{A}$ is the set of substrings that are NE of a particular type in $\mathbf{w}$.
FT	FSA, each for one type	Number of FT (of a particular type) in $\mathbf{w}$.
NW	SVM classifier	Score of SVM classifier.

It should be noted that different class models are constructed in different ways (e.g., NE models are *n*-gram models trained on corpora, whereas FT models use derivation rules and have binary values). The dynamic value ranges of different class model probabilities can be so different (some are not *probabilities* but *scores*) that it is inappropriate to combine all models through simple multiplication as in Equation (1). One way to balance these score quantities is to introduce for each class model (i.e., channel model) a model weight λ to adjust the class model score $P(\mathbf{s}|\mathbf{w})$ to $P(\mathbf{s}|\mathbf{w})^{\lambda}$. In our experiments, these weights are optimized so as to minimize the number of word segmentation errors on training data under the framework of linear models, as described in Section 4.2. It is worth noticing that the source–channel models are the rationale behind our system, e.g., the decoding process described in Section 5.6 follows the framework. Linear models are just another representation based on the optimization algorithm of class model weights.

4.2 Linear Models

The framework of linear models is derived from linear discriminant functions widely used for pattern classification (Duda, Hart, and Stork 2001) and has been recently introduced into NLP tasks by Collins and Duffy (2001). It is also related to (log-)linear models described in Berger, Della Pietra, and Della Pietra (1996), Xue (2003); Och (2003), and Peng, Feng, and McCallum (2004).

We use the following notation in the rest of the article.

- Training data are a set of example input/output pairs. In Chinese word segmentation, we have training samples $\{\mathbf{s}_i, \mathbf{w}_i^R\}$, for $i = 1 \ldots M$, where each $\mathbf{s}_i$ is an input Chinese character sequence and each $\mathbf{w}_i^R$ is the reference segmentation (i.e., word class sequence) of $\mathbf{s}_i$.
- We assume a set of $D + 1$ features $f_d(\mathbf{s}, \mathbf{w})$, for $d = 0 \ldots D$. The features are arbitrary functions that map $(\mathbf{s}, \mathbf{w})$ to real values. Using vector notation, we have $\mathbf{f}(\mathbf{s}, \mathbf{w}) \in \Re^{D+1}$, where $\mathbf{f}(\mathbf{s}, \mathbf{w}) = \{f_0(\mathbf{s}, \mathbf{w}), f_1(\mathbf{s}, \mathbf{w}), \ldots, f_D(\mathbf{s}, \mathbf{w})\}$. As shown in Table 6, $f_0(\mathbf{w})$ is called the base feature and is defined as the logarithm probability of the context model (i.e., word class trigram model). $f_d(\mathbf{s}, \mathbf{w})$, for $d = 1 \ldots D$, are defined for D word classes, respectively (i.e.,

they are basically derived from class models). Their values are either the summation of the logarithm of the probabilities of the corresponding probabilistic models, or assigned heuristically. Here, for those features that are only defined on $\mathbf{w}$, we will omit $\mathbf{s}$ and denote them as $f(\mathbf{w})$.

- Finally, the parameters of the model are a vector of $D + 1$ parameters, each for one feature function, $\boldsymbol{\lambda} = (\lambda_0, \lambda_1, \ldots, \lambda_D)$. $\lambda_1, \ldots, \lambda_D$ are in fact class model weights, described in Section 4.1. The likelihood score of a word class sequence $\mathbf{w}$ can be written as

$$Score(\mathbf{w}, \mathbf{s}, \boldsymbol{\lambda}) = \boldsymbol{\lambda}\mathbf{f}(\mathbf{w}, \mathbf{s}) = \sum_{d=0}^{D} \lambda_d f_d(\mathbf{w}, \mathbf{s}) \cdot \tag{2}$$

We see that Equation (2) is yet another representation of the source–channel models described in Section 4.1 by introducing class weights (i.e., adjust $P(\mathbf{s}|\mathbf{w})$ to $P(\mathbf{s}|\mathbf{w})^{\lambda}$) and taking the logarithm of all probabilities. The decision rule of Equation (1) can then be rewritten as

$$\mathbf{w}^* = \arg\max_{\mathbf{w} \in \mathbf{GEN(s)}} Score(\mathbf{w}, \mathbf{s}, \boldsymbol{\lambda}) \cdot \tag{3}$$

In what follows, we will describe the way of estimating λ under the framework of gradient descent: an iterative procedure of adjusting the parameters of λ in the direction that minimizes the segmentation errors with respect to a loss function. We will present in turn the loss function and the optimization algorithm.

4.2.1 Loss Function. Assume that we can measure the number of segmentation errors in $\mathbf{w}$ by comparing it with a reference segmentation $\mathbf{w}^R$ using an error function $Er(\mathbf{w}^R, \mathbf{w})$ (i.e., editing distance, in our case). The training criterion that directly minimizes the segmentation errors over the training data is

$$\boldsymbol{\lambda}^* = \arg\min_{\boldsymbol{\lambda}} \sum_{i=1 \ldots M} Er(\mathbf{w}_i^R, \mathbf{w}(\mathbf{s}_i, \boldsymbol{\lambda})), \tag{4}$$

where $\mathbf{w}(\mathbf{s}_i, \boldsymbol{\lambda})$ is the segmentation determined by Equation (3), where it is denoted as $\mathbf{w}^*$. Equation (4) is referred to as the minimum sample risk (MSR; Gao et al. 2005) criterion hereafter. Notice that without knowing the "true" distribution of the data, the best $\boldsymbol{\lambda}$ can be chosen approximately based on training samples. This is known as the principle of empirical risk minimization (ERM; Vapnik 1998): If the segmenter were trained using exactly the MSR criterion, it would converge to a Bayes risk performance (minimal error rate) as the training size goes to infinity.

However, $Er(.)$ is a piecewise constant function of the model parameter $\boldsymbol{\lambda}$, and thus a poor candidate for optimization by any simple gradient-type numerical search. For example, the gradient cannot be computed explicitly because $Er(.)$ is not differentiable with respect to $\boldsymbol{\lambda}$, and there are many local minima on the error surface. Therefore, we use an alternative loss function, minimum squared error (MSE) in equation (5), where $Score(.)$ is defined in Equation (2), where $\mathbf{s}$ has been suppressed

for simplicity. MSELoss($\boldsymbol{\lambda}$) is simply the squared difference between the score of the correct segmentation and the score of the incorrect one, summing over all training samples.

$$\text{MSELoss}(\boldsymbol{\lambda}) = \sum_{i=1\ldots M} (Score(\mathbf{w}_i^R, \boldsymbol{\lambda}) - Score(\mathbf{w}_i, \boldsymbol{\lambda}))^2 \cdot \quad (5)$$

It is useful to note that the MSE solution, under certain conditions, leads to approximations to maximum likelihood solution. The quality of the approximation depends upon the form of the linear discriminant functions (e.g., Equation (2)). Due to its appealing theoretical properties, the MSE criterion has received considerable attention in the literature, and there are many solution procedures available (Duda, Hart, and Stork 2001).

4.2.2 Optimization Algorithm. This section discusses the *delta rule*, a training algorithm for an unthresholded perceptron, following the description in Mitchell (1997). The delta rule in its component form is

$$\lambda_d = \lambda_d - \eta G(\lambda_d), \quad (6)$$

where η is the step size, and G is the gradient of MSELoss.[6] G can be estimated by differentiating the loss function of equation (7) with respect to λ_d as

$$G(\lambda_d) = \frac{\partial \text{MSELoss}(\lambda)}{\partial \lambda_d} = \sum_{i=1\ldots M} (Score(\mathbf{w}_i^R, \boldsymbol{\lambda}) - Score(\mathbf{w}_i, \boldsymbol{\lambda}))(f_d(\mathbf{w}_i^R) - f_d(\mathbf{w}_i)) \cdot \quad (7)$$

However, the objective function of Equation (5) in the context of our task (i.e., Chinese word segmentation) has many local minima, and thus gradient descent cannot guarantee finding the global minimum. We therefore use a *stochastic approximation* to gradient descent. Whereas the gradient descent computes parameter updates after summing over all training samples as shown in Equation (7), the stochastic approximation method updates parameters incrementally, following the calculation of the error for each individual training sample, as shown in Equation (8).

$$G(\lambda_d) = (Score(\mathbf{w}_i^R, \boldsymbol{\lambda}) - Score(\mathbf{w}_i, \boldsymbol{\lambda}))(f_d(\mathbf{w}_i^R) - f_d(\mathbf{w}_i)) \quad (8)$$

The stochastic approximation method can be viewed as optimizing a distinct loss function $\text{MSELoss}_i(\boldsymbol{\lambda})$ defined for each individual training sample i as follows

$$\text{MSELoss}_i(\boldsymbol{\lambda}) = (Score(\mathbf{w}_i^R, \boldsymbol{\lambda}) - Score(\mathbf{w}_i, \boldsymbol{\lambda}))^2 \cdot \quad (9)$$

6 η is usually taken as a time-decreasing function (e.g., Su and Lee 1994) to have a fast convergence speed in the beginning and a small variance in final iterations. We have tested this approach in our study, and it resulted in very limited improvement.

The optimization algorithm we used in our experiments is shown in Figure 4. It takes T passes over the training set. All parameters are initially set to be 1. The context model parameter λ_0 does change during training. Class model parameters are updated in a simple additive fashion: Parameters are altered according to the gradient with respect to $\text{MSELoss}_i(\boldsymbol{\lambda})$.

In our implementation, we approximate $\mathbf{w}^R$ as the $\mathbf{w}$ in $\mathbf{GEN}(\mathbf{s})$ with the fewest errors, so $Score(\boldsymbol{\lambda}, \mathbf{s}, \mathbf{w}^R) - Score(\boldsymbol{\lambda}, \mathbf{s}, \mathbf{w}) \leq 0$, where the equality holds only when $\mathbf{w}^R = \mathbf{w}$. That is, the model parameters are updated when the sentence is wrongly segmented. The update rule increases the parameter values for word classes whose models were "underestimated" (i.e., expected feature value $f(\mathbf{s}, \mathbf{w})$ is less than observed feature value $f(\mathbf{s}, \mathbf{w}^R)$), and decreases the parameter values whose models were "overestimated" (i.e., $f(\mathbf{s}, \mathbf{w})$ is larger than $f(\mathbf{s}, \mathbf{w}^R)$). Empirically, the sequence of these updates, when iterated over all training samples, provides a reasonable approximation to descending the gradient with respect to the original loss function of Equation (5). Although this method cannot guarantee a globally optimal solution, it is chosen for our modeling because of its efficiency and because it achieved the best results in our experiments.

The algorithm is similar to the perceptron algorithm described in Collins (2002). The key difference is that, instead of using the delta rule of Equation (8) (as shown in line 5 of Figure 4), Collins (2002) updates parameters using the rule: $\lambda_d^{t+1} \leftarrow \lambda_d^t + f_d(\mathbf{w}_i^R) - f_d(\mathbf{w}_i)$. Our pilot study shows that our algorithm achieves slightly better results.

4.3 Discussions on Robustness

The training methods described in Section 4.2 aim at minimizing errors in a training set. But test sets can be different. The *robustness* issue concerns how well the minimal error rate in the training set preserves in the test set. According to Dudewicz and Mishra (1988), the MSE function in general is not very robust because it is not bounded and can be contaminated from those training samples far away from the decision boundary. One of many possible solutions for improving the robustness is to introduce a *margin* in the training procedure of Figure 4. The basic idea is to enlarge the score difference (or score margin) between a correct segmentation (i.e., $\mathbf{w}^R$) and its competing incorrect segmentations (i.e., $\{\mathbf{w}; \mathbf{w} \in \mathbf{GEN}(\mathbf{s}), \mathbf{w} \neq \mathbf{w}^R\}$). According to Equation (8), the perceptron training algorithm of Figure 4 does not adjust parameters if the sentence is segmented correctly. The robustness could be improved if we continued to enlarge the score margin between the correct segmentation and the top competing candidate even if the input sentence had been correctly segmented, until the

ALGORITHM: PERCEPTRON-TRAINING

Input: Training samples $\{\mathbf{s}_i, \mathbf{w}_i^R\}$, for $i = 1 \ldots M$, with feature vectors $\mathbf{f}(\mathbf{s}, \mathbf{w}^R)$. # of iteration round T.
Initialization: $\lambda_i = 1$, $i = 0, 1, \ldots, M$.
1. **for** $t \leftarrow 1$ **to** T
2. **for** $i \leftarrow 1$ **to** M
3. $\mathbf{w}_i = \text{argmax}_{\mathbf{w} \in \text{GEN}(\mathbf{s}_i)} \boldsymbol{\lambda}^{t-1} \mathbf{f}(\mathbf{s}, \mathbf{w})$
4. **for** $d \leftarrow 1$ **to** D
5. $\lambda_d^t \leftarrow \lambda_d^{t-1} - \eta G(\lambda_d^{t-1})$, where $G(\lambda_d^{t-1})$ is calculated by Equation (8), and $\eta = 0.001$.
Output: Final parameter setting $\boldsymbol{\lambda}^T$

Figure 4
The perceptron training algorithm for Chinese word segmentation.

margin has exceeded a preset threshold. More specifically, we can modify Equation (8) as follows

$$G(\lambda_d) = (Score(\mathbf{w}_i^R, \boldsymbol{\lambda}) - Score(\mathbf{w}_i, \boldsymbol{\lambda}) - \delta)(f_d(\mathbf{w}_i^R) - f_d(\mathbf{w}_i)), \quad (10)$$

where δ is the desired margin that can either be an absolute value or a quantity proportional to the score of the correct segmentation (Su and Lee 1994). The modified training algorithm is similar to the *perceptron algorithm with margins* proposed by Krauth and Mèzard (1987). We leave the evaluation of the algorithm to future work.

Readers can also refer to Duda, Hart, and Stork (2001) for a detailed description of margin-based learning algorithms.

5. System Description

5.1 Architecture Overview

MSRSeg consists of two components: a generic segmenter and a set of output adaptors. We describe the first component in this section and leave the second to Section 6. The generic segmenter has been developed on the basis of the mathematical models described in Section 4. It consists of several components, as shown in Figure 5.

1. **Sentence Segmenter:** To detect sentence boundaries using punctuation clues such as . , ? and !.
2. **Word Candidate Generator:** Given an input string **s**, to generate all word candidates and store them in a word lattice. Each candidate is assigned to its word class and the class model score, e.g., $\log P(\mathbf{s}'|\mathbf{w})$, where **s**′ is any substring of **s**.

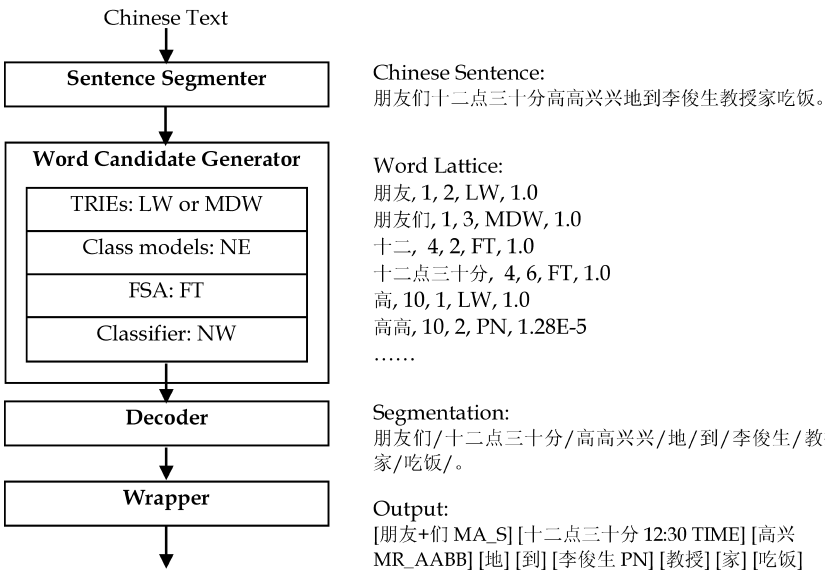


Figure 5
Overall architecture of MSRSeg.

3. **Decoder:** To select the best (or the *N* best) word segmentation (i.e., word class sequence **w***) from the lattice according to Equations (4) and (5), using the Viterbi (or A*) search algorithm.

4. **Wrapper:** To output segmentation results using some predefined canonical forms, e.g., MDW and FT are of their normalized form, as described in Section 3.1.

Here, we will describe candidate generators for different word classes. They are (1) the lexicon (and morph-lexicon) TRIEs to generate LW (or MDW) candidates, (2) the NE class models to generate NE candidates, (3) the finite-state automaton (FSA) to generate FT candidates, and (4) the classifier to generate NW candidates.

5.2 Lexicon Representation and Morphological Analysis

Lexicon words are represented as a set of TRIEs (Frakes and Baeza-Yates 1992), which is a particular implementation of the FSA described in Section 4. Given a character string, all prefix strings that form lexical words can be retrieved by browsing the TRIE whose root represents its first character.

Though there are well-known techniques for English morphological analysis (e.g., finite-state morphology), they are difficult to extend to Chinese for two reasons. First, Chinese morphological rules are not as *general* as their English counterparts. For example, in most cases English plural nouns can be generated using the rule "noun + s → plural noun". But only a small subset of Chinese nouns can be pluralized (e.g., 朋友们 'friends') using its Chinese counterpart "noun + 们 → plural noun" whereas others (e.g., 南瓜 'pumpkins') cannot.[7] Secondly, the operations required by Chinese morphological analysis, such as copying in reduplication, merging, and splitting, cannot be implemented using current finite-state networks.[8]

Our solution is extended lexicalization. We simply collect all MDWs of the five types described in Section 3.1 and incorporate them into the TRIE lexicon, called *morph-lexicon*. The TRIEs are essentially the same as those used for lexical words, except that not only the MDW's identity but also its morphological pattern and stem(s) are stored.

The procedure of lexicalization involves three steps: **(1) Candidate generation.** Candidate generation is done by applying a set of morphological rules to both the word lexicon and a large corpus. For example, the rule 'noun + 们 → plural noun' would generate candidates like 朋友们. **(2) Statistical filtering.** For each candidate, we obtain a set of statistical features such as frequency, mutual information, and left/right context dependency from a large corpus. We then use an information-gain-like metric described in Chien (1997) and Gao et al. (2002) to estimate how likely a candidate is to form a morphologically derived word and remove the "bad" candidates. The basic idea behind the metric is that a Chinese word should appear as a stable sequence in the corpus. That is, the components within the word are strongly correlated, while the components at both ends should have low correlations with words outside the sequence. **(3) Linguistic selection.** Finally, we manually check the remaining candidates

7 We do not consider those special cases, such as in a children's fairy story, where the magic pumpkins can talk.

8 There are some types of copying operations that can be implemented by FSMs (e.g., Sproat 1992), but implementation is not trivial. Because the number of MDWs is manageable, storing them as a list is not much more expensive than storing them as a finite-state network, in terms of space and access speed. Our implementation can be viewed as a pragmatic solution, easy to implement and maintain.

and construct the morph-lexicon, where each entry is tagged with its morphological pattern and stem(s). The resulting morph-lexicon contains 50,963 MDWs.

5.3 Named Entities

We consider four types of named entities: person names (PNs), location names (LNs), organization names (ONs), and transliterations of foreign names (FNs). Because any character string can in principle be a named entity of one or more types, in order to limit the number of candidates for a more effective search, we generate named entity candidates given an input string in two steps: First, for each type, we use a set of constraints (which are compiled by linguists and are represented as FSAs) to generate only those "most likely" candidates. Second, each of the generated candidates is assigned a class model probability. Class models are defined as generative models that are estimated on their corresponding named entity lists using MLE, together with a backoff smoothing schema, as described in Section 4.1.1. We will describe briefly the constraints and the class models here. The Chinese person-name model is a modified version of that described in Sproat et al. (1996). Other NE models are novel, though they share some similarities with the Chinese person-name model.

5.3.1 Chinese Person Names. There are two main constraints. (1) PN patterns: We assume that a Chinese PN consists of a family name **F** and a given name **G**, and is of the pattern **F**+**G**. Both **F** and **G** are one or two characters long. (2) Family name list: We only consider PN candidates that begin with an **F** stored in the family name list (which contains 373 entries in our system).

Given a PN candidate, which is a character string **s**, the class model probability $P(\mathbf{s}|\text{PN})$ is computed by a character bigram model as follows: (1) generate the family name substring $\mathbf{s}_F$, with the probability $P(\mathbf{s}_F|\text{F})$; (2) generate the given name sub-string $\mathbf{s}_G$, with the probability $P(\mathbf{s}_G|\text{G})$(or $P(\mathbf{s}_{G1}|\text{G1})$); and (3) generate the second given name, with the probability $P(\mathbf{s}_{G2}|\mathbf{s}_{G1}, \text{G2})$. For example, the generative probability of the string 李俊生 given that it is a PN would be estimated as P(李俊生|PN) = P(李|F)P(俊|G1)P(生|俊, G2).

5.3.2 Location Names. Unlike PNs, there are no patterns for LNs. We assume that an LN candidate is generated given **s** (which is less than 10 characters long), if one of the following conditions is satisfied: (1) **s** is an entry in the LN list (which contains 30,000 LNs); (2) **s** ends in a keyword in a 120-entry LN keyword list (e.g., 市 'city').[9]

The probability $P(\mathbf{s}|\text{LN})$ is computed by a character bigram model. Consider a string 夏米尔河 'Shamir river'. It is an LN candidate because it ends in an LN keyword 河 'river'. The generative probability of the string, given that it is an LN, would be estimated as P(夏米尔河|LN) = P(夏|<LN>) P(米|夏) P(尔|米) P(河|尔) P(</LN>|河), where <LN> and </LN> are symbols denoting the beginning and the end of an LN, respectively.

5.3.3 Organization Names. ONs are more difficult to identify than PNs and LNs because ONs are usually nested named entities. For example, the ON 中国国际航空公司 'Air China Corporation' contains an LN 中国 'China'.

9 For clarity, the constraint is a simplified version of that used in MSRSeg.

Like the identification of LNs, an ON candidate is only generated given a character string **s** (less than 15 characters long), if it ends in a keyword in a 1,355-entry ON keyword list (e.g., 公司 'corporation'). To estimate the generative probability of a nested ON, we introduce word class segmentations of **s, w,** as hidden variables. In principle, the ON class model recovers $P(\mathbf{s}|\text{ON})$ over all possible C: $P(\mathbf{s}\,|\,\text{ON}) = \Sigma_{\mathbf{w}} P(\mathbf{s},\mathbf{w}\,|\,\text{ON}) = \Sigma_{\mathbf{w}} P(\mathbf{w}\,|\,\text{ON}) P(\mathbf{s}\,|\,\mathbf{w},\text{ON})$. Since $P(\mathbf{s}\,|\,\mathbf{w},\text{ON}) = P(\mathbf{s}\,|\,\mathbf{w})$, we have $P(\mathbf{s}\,|\,\text{ON}) = \Sigma_{\mathbf{w}} P(\mathbf{w}\,|\,\text{ON}) P(\mathbf{s}\,|\,\mathbf{w})$. We then assume that the sum is approximated by a single pair of terms $P(\mathbf{w}^*|\,\text{ON}) P(\mathbf{s}\,|\,\mathbf{w}^*)$, where $\mathbf{w}^*$ is the most probable word class segmentation discovered by Equation (5). That is, we also use our system to find **w***, but the source–channel models are estimated on the ON list.

Consider the earlier example. Assuming that **w*** = LN/国际/航空/公司, where 中国 is tagged as an LN, the probability $P(\mathbf{s}\,|\,\text{ON})$ would be estimated using a word class bigram model as: $P(\text{中国国际航空公司}\,|\,\text{ON}) \approx P(\text{LN/国际/航空/公司}\,|\,\text{ON})$ $P(\text{中国}\,|\,\text{LN}) = P(\text{LN}\,|\,\text{<ON>})\, P(\text{国际}\,|\,\text{LN})\, P(\text{航空}\,|\,\text{国际})\, P(\text{公司}\,|\,\text{航空})\, P(\text{</ON>}\,|\,\text{公司})$ $P(\text{中国}\,|\,\text{LN})$, where $P(\text{中国}\,|\,\text{LN})$ is the class model probability of 中国 given that it is an LN, and <ON> and </ON> are symbols denoting the beginning and the end of an ON, respectively.

5.3.4 Transliterations of Foreign Names. As described in Sproat et al. (1996), FNs are usually transliterated using Chinese character strings whose sequential pronunciation mimics the source language pronunciation of the name. Since FNs can be of any length and their original pronunciation is effectively unlimited, the recognition of such names can be tricky. Fortunately, there are only a few hundred Chinese characters that are particularly common in transliterations.

Therefore, an FN candidate would be generated given **s** if it contains only characters stored in a transliterated name character list (which contains 618 Chinese characters). The probability $P(\mathbf{s}|\text{FN})$ is estimated using a character bigram model. Notice that in our system an FN can be a PN, an LN, or an ON, depending on the context. Then, given an FN candidate, three named entity candidates, each for one category, are generated in the lattice, with the class probabilities $P(\mathbf{s}|\text{PN}) = P(\mathbf{s}|\text{LN}) = P(\mathbf{s}|\text{ON}) = P(\mathbf{s}|\text{FN})$. In other words, we postpone the determination of its type to the decoding phase where the context model is used.

5.3.5 Abbreviations. For the sake of completeness, we describe below the basic ideas for tackling NE abbreviations within our framework. This is ongoing research, and we do not have conclusive results yet. Readers can refer to Zhu et al. (2003) and Sun, Zhou, and Gao (2003) for more details, where marginal improvements have been reported.

For different types of NEs, different strategies can be adopted. We find that most abbreviations of Chinese PNs and LNs are single-character NEs. The PN and LN models described previously cannot handle them very well because (1) single-character NEs are generated in a different way from that of multicharacter NEs, and (2) the context of single-character NEs is different from multicharacter ones. For example, single-character NEs usually appear adjacent to one another, such as 中 and 俄 in 中俄贸易 'China-Russia trade'. But this is not the case for multicharacter NEs.

We thus define single-character PNs and LNs as two separate word classes, denoted by SCPN and SCLN, respectively. We assume that a character is a candidate of SCPN (or SCLN) only when it is included in a predefined SCPN (or SCLN) list, which

contains 151 (or 177) characters. The class model probabilities are assigned by unigram models as

$$P(s \mid \text{SCPN}) = \frac{C(s)}{\sum_{i=1\ldots N} C(s_i)}, P(s \mid \text{SCLN}) = \frac{C(s)}{\sum_{i=1\ldots N} C(s_i)}, \quad (11)$$

where C(s) is the count of the SCPN (or SCLN) s in an annotated training set and N is the size of the SCPN (or SCLN) list. In the context model, we also differentiate between PN (or LN) and SCPN (or SCLN). Therefore, SCPN and SCLN should be labeled explicitly in the training data.

It is much more difficult to detect abbreviations of ONs (denoted by ONA) because ONAs are usually multiple-character strings and can be generated from their original ON arbitrarily. For example, the abbreviation of 清华大学 'Tsinghua University' is 清华 while the abbreviation of 北京大学 'Peking University' is 北大. We assume that an ONA candidate is only generated given a character string **s** (fewer than 6 characters long), if both of the following conditions are satisfied: (1) An ON has been detected in the same document, and (2) **s** can be derived from the ON using a generative pattern defined in Table 7. Since there is no training data for the ONA class model, we construct a score function to assign each ONA candidate a class model score. Consider a string 北大. The generative probability of the string, given it is an ONA, would be approximated as P(北大|ONA) $\approx$ P(北京大学|ON) P(北大|北京大学, ON), where P(北大|北京大学, ON) is defined as a constant (0.8 in our experiments) if 北大 can be derived from 北京大学 using any generative pattern of Figure 7; 0, otherwise. If more than one ON is detected previously in the same document and can be used to derive the ONA candidate (e.g., 北方大学 'North University'), only the closest ON is taken into account. We also notice that ONAs and ONs occur in similar contexts. So we do not differentiate them in the context model.

5.4 Factoids

The types of factoids handled in MSRSeg are shown in Table 1 of Section 3.1. For each type of factoid, we generate a grammar of regular expressions. The ten regular expressions are then compiled into a single FSA. Given an input string **s,** we scan it from left to right, and output a FT candidate when a substring matches one of the ten

Table 7
Generative patterns of ONA, where s_{ij} denotes the j-th character of the i-th word of ON (Sun, Zhou and Gao 2003).

Condition (ON)	Generated patterns (ONA)	Examples
ON $= w_1 \ldots w_N$ ($N \geq 2$)	ONA $= s_{11}s_{21}$, or	北京/邮电/大学 → 北邮
	ONA $= s_{11}s_{21}s_{31}$, or ...	北京/邮电/大学 → 北邮大
	ONA $= s_{11}s_{21}\ldots s_{N1}$	
ON $= w_1w_2$, and w_1 is not an LN	ONA $= w_1$	清华/大学 → 清华
ON $= w_1w_2w_3$, and w_1 is not an LN	ONA $= w_1$, *or*	苹果/电脑/公司 → 苹果
	ONA $= w_1w_2$	苹果/电脑/公司 → 苹果/电脑
ON $= w_1w_2w_3$, and w_1 is an LN	ONA $= w_2$	北京/国安/队 → 国安

regular expressions. We also remove those FT candidates that are substrings of any other candidates of the same type. Consider the example in Figure 1; only the string 十二点三十分 is accepted as a FT candidate (a time expression), not the substrings (e.g., 十二点 or 十二点三十).

The use of FSA is motivated by the fact that the detection of most FTs is based exclusively on their internal properties and without relying on context. This can be in principle justified by experiments. As shown in Table 8, the overall performance of FT detection using only FSA is comparable with that of using MSRSeg where the contextual information of the FT is considered (i.e., in MSRSeg, FSA are used as feature functions, and the FT are detected simultaneously with other words). If we read the results carefully, we can see that the use of context information (in MSRSeg) achieves a higher precision but a lower recall—a small but significant difference.

5.5 New Words

New words in this section refer to OOV words that are neither recognized as named entities or factoids nor derived by morphological rules. These words are mostly domain-specific and/or time-sensitive, such as 三通 'Three Links', 非典 'SARS'. The identification of such new words has not been studied extensively before. It is an important issue that has substantial impact on the performance of word segmentation. For example, approximately 30% of OOV words in the SIGHAN's PK corpus in Table 4 are new words of this type. There has been previous work on detecting Chinese new words from a large corpus in an off-line manner and updating the dictionary before word segmentation. However, our approach is able to detect new words on-line, i.e., to spot new words in a sentence on the fly during the process of word segmentation, where widely used statistical features such as mutual information or term frequency are not available.

For brevity, we will focus on the identification of two-character new words, denoted as NW_11. Other types of new words such as NW_21 (a two-character word followed with a character) and NW_12 can be detected similarly (e.g., by viewing the two-character word as an inseparable unit, like a character). These three types amount to 85% of all NWs in the PK corpus. Here, we shall describe the class model and context model for NWI, and the creation of training data by sampling.

5.5.1 Class Model. We use a classifier (SVM^{light} [Joachims 2002] in our experiments) to estimate the likelihood of two adjacent characters forming a new word. Of the great number of features with which we experimented, four linguistically motivated features are chosen due to their effectiveness and availability for on-line detection. They are *independent word probability* (IWP), *anti-word pair* (AWP), *word formation analogy*

Table 8
FT detection results on the MSR gold test set. The 'All' column shows the results of detecting all 10 types of factoids, as described in Table 1, which amount to 6630 factoids, as shown in Table 3.

	All (P/R)	dat (P/R)	tim (P/R)	mea (P/R)	mon (P/R)
FSA	**0.906/0.905**	0.967/0.870	0.967/0.967	0.911/0.919	0.991/0.975
MSRSeg	**0.908/0.896**	0.980/0.851	0.973/0.967	0.913/0.898	0.994/0.975

(WFA), and *morphological productivity* (MP). We now describe each feature in turn. In Section 5.5.2, we shall describe the way the training data (new word list) for the classifier is created by sampling.

IWP (independent word probability) is a real-valued feature. Most Chinese characters can be used either as independent words or component parts of multicharacter words, or both. The IWP of a single character is the likelihood of this character to appear as an independent word in texts (Wu and Jiang 2000):

$$IWP(x) = \frac{C(x, W)}{C(x)}, \tag{12}$$

where $C(x, W)$ is the number of occurrences of the character x as an independent word in training data, and $C(x)$ is the total number of occurrences of x in training data. We assume that the IWP of a character string is the product of the IWPs of the component characters. Intuitively, the lower the IWP value, the more likely the character string forms a new word. In our implementation, the training data is word-segmented.

AWP (anti-word pair) is a binary feature derived from IWP. For example, the value of AWP of an NW_11 candidate ab is defined as: $AWP(ab)$=1 if $IWP(a)>\theta$ or $IWP(b) >\theta$; 0, otherwise. $\theta \in [0, 1]$ is a preset threshold. Intuitively, if one of the component characters is very likely to be an independent word, it is unlikely to be able to form a word with any other characters. While IWP considers all component characters in a new word candidate, AWP only considers the one with the maximal IWP value.

WFA (word formation analogy) is a binary feature. Given a character pair (x, y), a character (or a multicharacter string) z is called the common stem of (x, y) if at least one of the following two conditions hold: (1) character strings xz and yz are lexical words (i.e., x and y as prefixes); and (2) character strings zx and zy are lexical words (i.e., x and y as suffixes). We then collect a list of such character pairs, called affix pairs, of which the number of common stems is larger than a preset threshold. The value of WFA for a given NW_11 candidate ab is defined as: $WFA(ab) = 1$ if there exists an affix pair (a, x) (or (b, x)) and the string xb (or ax) is a lexical word; 0, otherwise. For example, given an NW_11 candidate 下岗 (xia4-gang3, 'be laid off'), we have WFA(下岗) = 1 because (上, 下) is an affix pair (they have 32 common stems such as _任, _游, _台, _车, 面, _午, _班) and 上岗 (shang4-gang3, 'take over a shift') is a lexical word.

MP (morphological productivity) is a real-valued feature. It is a measure of the productivity of a particular construction, as defined here (Baayen 1989):

$$MP(x) = \frac{n_1(x)}{N(x)}. \tag{13}$$

MP is strongly related to the Good-Turing estimate. Here, N is the number of tokens of a particular construction found in a corpus, e.g., the number of tokens of all nouns ending in -们, and n_1 is the number of types of that construction, e.g., the number of unique nouns ending in -们. Intuitively, a higher value of MP indicates a higher probability that (one of) the component parts of a multicharacter string appears to be a word. For example, Sproat and Shih (2002) show that the MP values of Chinese noun affix -们 and verb affix -过 are 0.20 and 0.04, respectively, indicating that -们 is a much more productive affix, while the MP value of single-character nouns which belong to

a closed and nonproductive class is close to 0. These results are in agreement with our intuition. Similarly, we find some very productive characters with high MP values. For example, in our training set, there are 236 words that contain the character 帽, among which 13 occur only once.

5.5.2 Context Model. The motivations for using a context model for NWI are twofold. The first is to capture useful contextual information. For example, new words are more likely to be nouns than pronouns, and POS tagging is context-sensitive. The second is more important. As described in Section 4, with a context model, NWI can be performed simultaneously with other word segmentation tasks (e.g., word breaking, NER, and morphological analysis) in a unified manner.

However, it is difficult to develop a training corpus where new words are annotated because "we usually do not know what we don't know." Our solution is Monte Carlo simulation. We sample a set of new words from our dictionary according to the distribution—the probability that any lexical word w would be a new word $P(NW|w)$. We then generate a new-word-annotated corpus from a word-segmented text corpus.

Now we describe the way $P(NW|w)$ is estimated. It is reasonable to assume that new words are those words whose probability of appearing in a new document is lower than general lexical words. Let $P_i(k)$ be the probability of word w_i that occurs k times in a document. In our experiments, we assume that $P(NW|w_i)$ can be approximated by the probability of w_i occurring less than K times in a new document:

$$P(NW \mid w_i) \approx \sum_{k=0}^{K\,1} P_i(k), \tag{14}$$

where the constant K (7 in our experiments) is dependent on the size of the document: The larger the document, the larger the value. $P_i(k)$ can be estimated using several term distribution models (Chapter 15.3 in Manning and Schütze [1999]). Following Gao and Lee (2000), we use K-Mixture (Katz 1996) which estimates $P_i(k)$ as

$$P_i(k) = (1 - a)\delta_{k,\,0} + \frac{a}{\beta + 1}\left(\frac{\beta}{\beta + 1}\right)^k, \tag{15}$$

where $\delta_{k,0}$=1 if k=0; 0, otherwise. α and β are parameters that can be fit using the observed mean λ and the observed inverse document frequency IDF as follows:

$$\lambda = \frac{cf}{N}, IDF = \log\frac{N}{df},$$

$$\beta = \lambda \times 2^{IDF} - 1 = \frac{cf - df}{df}, \text{ and } a = \frac{\lambda}{\beta}, \tag{16}$$

where cf is the total number of occurrence of word w_i in training data, df is the number of documents in training data in which w_i occurs, and N is the total number of documents. In our implementation, the training data contain approximately 40,000 documents that have been balanced among domain, style, and time.

Table 9
NW_11 identification results on PK test set.

	P	R	F
All features	0.565	0.788	0.658
w/o IWP	0.395	0.835	0.536
w/o AWP	0.508	0.723	0.596
w/o MP	0.556	0.771	0.646
w/o WFA	0.561	0.779	0.652

Table 10
NW_21 identification results on PK test set.

	P	R	F
All features	0.420	0.811	0.553
w/o IWP	0.104	0.973	0.188
w/o AWP	0.338	0.730	0.462
w/o MP	0.398	0.716	0.512
w/o WFA	0.401	0.797	0.534

5.5.3 Evaluation Results. The NWI component has been constructed as an SVM classifier. This section discusses two factors that we believe have the most impact on the performance of NWI. First, we investigate the relative contribution of the four linguistically motivated features in NWI. Second, we compare methods where we use the NWI component (i.e., an SVM classifier) as a post-processor versus as a feature function in the linear models of Equation (4).

Table 11
NWI results on PK and CTB corpora, NWI as post-processor versus unified approach.

	PK				CTB			
MSRSeg	Roov	F	P	R	Roov	F	P	R
w/o NWI	.741	.956	.949	.963	.690	.892	.875	.910
w/ NWI (post-processor)	.746	.953	.947	.960	.722	.899	.886	.912
w/ NWI (unified approach)	.781	.955	.952	.959	.746	.904	.895	.914

Table 12
NWI results on HK and AS corpora, NWI as post-processor versus unified approach.

	HK				AS			
MSRSeg	Roov	F	P	R	Roov	F	P	R
w/o NWI	.694	.947	.937	.958	.436	.951	.958	.943
w/ NWI (post-processor)	.728	.952	.944	.959	.549	.955	.950	.959
w/ NWI (unified approach)	.746	.954	.948	.960	.584	.958	.955	.961

The NWI results on the PK test set are shown in Tables 9 and 10. We turned off the features one at a time and recorded the scores of each ablated NWI component. It turns out that in cases of both NW_11 and NW_12, IWP is obviously the most effective feature.

Tables 11 and 12 show results of NWI on four Bakeoff test sets. We can see that unified approaches (i.e., using the NWI component as a feature function) significantly outperform consecutive approaches (i.e., using the NWI component as a post-processor) consistently, in terms of both Roov and P/R/F of the overall word segmentation. This demonstrates empirically the benefits of using the context model for NWI and the unified approach to Chinese word segmentation, as described in 5.5.2.

5.6 Decoder

The decoding process follows the source–channel framework. It consists of three steps:

Step 1: Throughout the process, we maintain an array of word class candidates, called a *lattice*, which is initialized to be empty.

Step 2: Given a Chinese sentence, all possible words of different types are generated simultaneously by the corresponding channel models described in Sections 5.2 to 5.5. For example, as shown in Table 6, the lexicon TRIE generates LW candidates; the SVM classifier generates NW candidates, and so on. All the generated candidates are added to the lattice. Each element in the lattice is a 5-tuple (w, i, l, t, s), where w is the word candidate, i is the starting position of w in the sentence, l is the length of w, t is the word class tag, and s is the class model score of w assigned by its feature function in Table 6. Some examples are shown in Figure 5.

Step 3: The Viterbi (or A*) algorithm is used to search for the best word class sequence, among all candidate segmentations in the lattice, according to Equations (2) and (3).

For efficiency, we sometimes need to control the search space. Given an input sentence **s,** all word candidates are ranked by their normalized class model score $\lambda f(.)$. Thus, the number of candidates (i.e., the size of the lattice) is controlled by two parameters:

- Number threshold: The maximum number of candidates cannot be larger than a given threshold;
- Score threshold: The difference between the class model score of the top-ranked candidates and the bottom-ranked candidates cannot be larger than a given threshold.

6. Standards Adaptation

This section describes the second component of MSRSeg: a set of adaptors for adjusting the output of the generic segmenter to different application-specific standards.

We consider the following standards adaptation paradigm. Suppose we have a *general* standard predefined by ourselves. We have also created a large amount of training data that are segmented according to this general standard. We then develop a *generic* word segmenter, i.e., the system described in Section 5. Whenever we

deploy the segmenter for any application, we need to customize the output of the segmenter according to an application-specific standard, which is not always explicitly defined. However, it is often implicitly defined in a given amount of application data (called adaptation data) from which the specific standard can be partially acquired.

6.1 Transformation-Based Learning Approach

In MSRSeg, standards adaptation is conducted by a postprocessor that performs an ordered list of transformations on the output of the generic segmenter—removing extraneous word boundaries and inserting new boundaries—to obtain a word segmentation that meets a different standard.

The method we use is Transformation-Based Learning (TBL; Brill [1995]), which requires an initial segmentation, a goal segmentation into which we wish to transform the initial segmentation, and a space of allowable transformations (i.e., transformation templates). Under the above-mentioned adaptation paradigm, the initial segmentation is the output of the generic segmenter. The goal segmentation is adaptation data. The transformation templates can make reference to words (i.e., lexicalized templates) as well as some predefined types (i.e., class-type based templates), as described below.

We notice that most variability in word segmentation across different standards comes from those words that are not typically stored in the dictionary. Those words are dynamic in nature and are usually formed through productive morphological processes. In this study, we focus on three categories: MDW, NE, and FT.

For each word class that belongs to these categories, we define an internal structure similar to Wu (2003). The structure is a tree with 'word class' as the root and 'component types' as the other nodes. There are 30 component types. As shown in Figure 6, the word class Affixation has three component types: Prefix, Stem, and Suffix. Similarly, PersonName has two component types and Date has nine—3 as non-terminals and 6 as terminals. These internal structures are assigned to words by the generic segmenter at

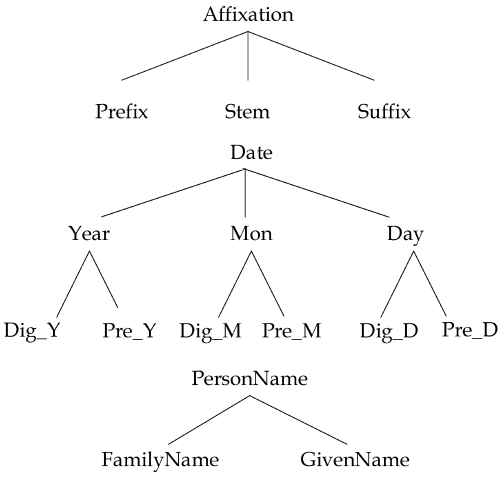


Figure 6
Word internal structure and class-type transformation templates.

Table 13
Comparison scores for PK open and CTB open.

	PK (# of Tr. Word = 1.1M)					CTB (# of Tr. Word = 250K)				
	P	R	F	Roov	Riv	P	R	F	Roov	Riv
1. MSRSeg w/o adaptation	.824	.854	.839	.320	.861	.799	.818	.809	.624	.861
2. MSRSeg	.952	.959	.955	.781	.972	.895	.914	.904	.746	.950
3. FMM w/ adaptation	.913	.946	.929	.524	.977	.805	.874	.838	.521	.952
4. Rank 1 in Bakeoff	.956	.963	.959	.799	.975	.907	.916	.912	.766	.949
5. Rank 2 in Bakeoff	.943	.963	.953	.743	.980	.891	.911	.901	.736	.949

run time. The transformation templates for words of the above three categories are of the form:

Condition: word class
Actions:

- Insert: Place a new boundary between two component types.
- Delete: Remove an existing boundary between two component types.

Since the application of the transformations derived from the above templates is conditioned on word class and makes reference to component types, we call the templates class-type transformation templates. Some examples are shown in Figure 6.

In addition, we also use lexicalized transformation templates as follows:

- Insert: Place a new boundary between two lemmas.
- Delete: Remove an existing boundary between two lemmas.

Here, *lemmas* refer to those basic lexical words that cannot be formed by any productive morphological process. They are mostly single characters, two-character words, and 4-character idioms.

6.2 Evaluation Results

The results of standards adaptation on four Bakeoff *open* test sets are shown in Tables 13 and 14.[10] A set of transformations for each standard is learned using TBL from the corresponding Bakeoff training set. For each test set, we report results using our system with and without standards adaptation (Rows 1 and 2). It turns out that performance improves dramatically across the board in all four test sets.

For comparison, we also include in each table the results of using the FMM (forward maximum matching) greedy segmenter as a generic segmenter (Row 3), and the top 2 scores (sorted by F) that are reported in SIGHAN's First International Chinese Word Segmentation Bakeoff (Rows 4 and 5).[11] We can see that with adaptation,

10 See Section 8.3 for the definitions of **open test** and **close test**.
11 The FMM algorithm processes through the sentence from left to right, taking the longest match with the lexicon entry at each point. Similarly, the BMM (backward maximum matching) algorithm processes the sentence from right to left.

Table 14
Comparison scores for HK open and AS open.

	HK (# of Tr. Word = 240K)					AS (# of Tr. Word = 5.8M)				
	P	R	F	Roov	Riv	P	R	F	Roov	Riv
1. MSRSeg w/o adaptation	.819	.822	.820	.593	.840	.832	.838	.835	.405	.847
2. MSRSeg	.948	.960	.954	.746	.977	.955	.961	.958	.584	.969
3. FMM w/ adaptation	.818	.823	.821	.591	.841	.930	.947	.939	.160	.964
4. Rank 1 in Bakeoff	.954	.958	.956	.788	.971	.894	.915	.904	.426	.926
5. Rank 2 in Bakeoff	.863	.909	.886	.579	.935	.853	.892	.872	.236	.906

Table 15
Size of training data set and the adaptation results on AS open.

Size of training set(# words)	R	P	F	Roov	Riv
5.8 M	.961	.956	.958	.603	.968
2.9 M	.959	.950	.954	.623	.966
1.1 M	.954	.945	.949	.591	.962
.25 M	.947	.935	.941	.595	.955
w/o adaptation	.832	.838	.835	.405	.847

our generic segmenter can achieve state-of-the-art performance on different standards, showing its superiority over other systems. For example, there is no single segmenter in the Bakeoff that achieved top-2 ranks in all four test sets (Sproat and Emerson 2003).

We notice in Tables 13 and 14 that the quality of adaptation seems to depend largely upon the size of adaptation data (indicated by # of Tr. Word in the tables): we outperformed the best Bakeoff systems in the AS set because of the large size of the adaptation data, while we are worse in the CTB set because of the small size of the adaptation data. To verify our hypothesis, we evaluated the adaptation results using subsets of the AS training set of different sizes and observed the same trend, as shown in Table 15. However, even with a much smaller adaptation data set (e.g., 250K words), we still outperform the best Bakeoff results.

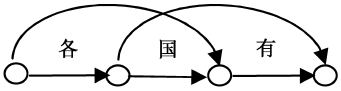
7. Training Data Creation

This section describes (semi-)automatic methods of creating the training data based on the estimated class model probability $P(\mathbf{w})$ (i.e., trigram probability) in Equation (1). Ideally, given an annotated corpus, where each sentence is segmented into words that are tagged by their classes, the trigram word class probabilities can be calculated using MLE. Unfortunately, building such annotated training corpora is very expensive.

7.1 Bootstrapping Approach and Beyond

Our basic solution is the bootstrapping approach described in Gao et al. (2002). It consists of three steps: (1) Initially, we use a greedy word segmenter to annotate the corpus and obtain an initial context model based on the initial annotated corpus; (2) we reannotate the corpus using the obtained models; and (3) we retrain the context

(a)



(b) 在/**各/国有**/企业/中 (in each state-owned enterprise)
在/人权/问题/上/,/**各国/有**/共同点
(Regarding human rights, every country has some common ground)

(c) 在/<OAS>/企业/中
在/人权/问题/上/,/<OAS>/共同点

Figure 7
(a) A Chinese OAS 各国有. (b) Two sentences in the training set, which contain the OAS and (c) whose OASs have been replaced with the single tokens <OAS>. (Li et al. 2003).

Table 16
Methods of resolving OAs in word segmentation, on the MSR test set.

Methods	Accuracy
FMM	73.1%
BMM	71.5%
Rule + FMM	90.7%
Rule + BMM	91.3%
Ours	94.3%

model using the reannotated corpus.[12] Steps 2 and 3 are iterated until the performance of the system converges. This approach is also named Viterbi iterative training, an approximation of EM training.

In the above approach, the quality of the context model depends to a large degree upon the quality of the initial annotated corpus, which is, however, not satisfied due to two problems. First, the greedy segmenter cannot deal with the segmentation ambiguities, and even after many iterations, these ambiguities can only be partially resolved. Second, many factoids and named entities cannot be identified using the greedy word segmenter, which is based on the dictionary.

To solve the first problem, we use two methods to resolve segmentation ambiguities in the initially segmented training data. We classify word segmentation ambiguities into two classes: overlap ambiguity (OA) and combination ambiguity (CA), corresponding, respectively, to OAS and CAS, defined in Section 3.5.

To resolve OA, we identify all OASs in the training data and replace them with a single token <OAS>. An example is shown in Figure 7. By doing so, we remove the portion of training data that are likely to contain OA errors. We thus train a context model using the reduced training set that does not contain any OASs. Intuitively, the resulting context model would resolve the ambiguities. The method has been tested on the MSR test set. Our main results are shown in Table 16. We can see that the FMM (or backward maximum matching—BMM) method can only resolve 73.1% (or 71.1%) of OAs, while using our method, the resulting context model can resolve 94.3% of the OAs. Our method is different from previous ones that use lexicalized rules to resolve OAS. For example, Sun and Zuo (1998) report that over 90% of OAs can be disambiguated

12 The greedy word segmenter is based on a FMM algorithm.

才能 与其 人为 着手 上将 将来 穿着 不过 成人 处在 到底 到手 得出 的话 上来 正当 中将 总会 等到 等同 等于 低下 地带 都会 多年 一度 一道 高等 上人 高地 个人 上前 过夜 和局 业已 回家 口水 后退 来使 中人 会同 中共 家当 马上 中和 口气 中游 家人 为人 教会 人手 人称 可靠 理事 全能 天下 在场 天才 市区 着眼 条约 应对 是非 正在 内在 名将 将就 同行 同一 使女	
Overall accuracy of Voting:	81.7%
Overall accuracy of ME:	94.1%
Overall accuracy of VSM:	**95.7%**

Figure 8
Results of 70 high-frequency two-character CASs. 'Voting' indicates the accuracy of the baseline method that always chooses the more frequent case of a given CAS. 'ME' indicates the accuracy of the maximum-entropy classifier. 'VSM' indicates the accuracy of the method of using VSM for disambiguation.

simply by rules. We reimplemented their method in our experiments and found that 90.7% (or 91.3%) of the OAs in the MSR test set can be resolved. The result is similar to Sun and Zou's but still not as good as ours. Therefore, we conclude that our method significantly outperforms the rule-based approaches. Another advantage of our method is that it is an unsupervised approach that requires no human annotation. Readers can refer to Li et al. (2003) for more details.

To resolve CA, we select 70 high-frequency two-character CASs (e.g., 才 能 'talent' and 才/能 'just able'), as shown in Figure 8. For each CAS, we train a binary classifier using sentences that contain the CAS and that have been segmented using the greedy segmenter. Then, for each occurrence of a CAS in the initial segmented training data, the corresponding classifier is used to determine whether the CAS should be segmented. Our experiments show that 95.7% of the CAs can be resolved. Detailed results are shown in Figure 8, where 'Voting' indicates the accuracy of the baseline method that always chooses the more frequent case of a given CAS, and 'VSM' indicates the accuracy of the VSM-inspired (vector space model) binary classifier, which will be described here.[13] Suppose we have a CAS, **s,** whose position in a sentence is i. We use its six surrounding words w in positions $i-3$, $i-2$, $i-1$, $i+1$, $i+2$, and $i+3$ as features. We then define a set of feature functions to simulate the TF-IDF scores. Each feature function is a mapping $f(\mathbf{s}, w) \in \Re$. In particular, let TF1(**s**, w) (or TF2(**s**, w)) be the term frequency of w in the case that **s** is a 1-word (or 2-word) string. Similarly, let IDF1 (**s**, w) (or IDF2 (**s**, w)) be 1 if w only occurs in the case that **s** is a 1-word (or 2-word) string. If w occurs in both cases, let IDF1 (**s**, w) = IDF2 (**s**, w) = 0.25. We also assign weight λ for each position empirically (i.e., in our experiments, we have λ_{-3}=λ_{+3}=1, λ_{-2}=λ_{+2}=4, and λ_{-1}=λ_{+1}=7). Then we can calculate the score of **s** to be 1-word or 2-word by Equations (18) and (19), respectively. The CAS is a single word if Score1(**s**) > Score2(**s**), and two words otherwise.

$$\text{Score 1}(\mathbf{s}) = \sum_{i=-3\ldots+3, i\neq 0} \text{TF 1}(\mathbf{s}, w_i)\text{IDF 1}(\mathbf{s}, w_i)\lambda_i \tag{17}$$

$$\text{Score 2}(\mathbf{s}) = \sum_{i=-3\ldots+3, i\neq 0} \text{TF 2}(\mathbf{s}, w_i)\text{IDF 2}(\mathbf{s}, w_i)\lambda_i \tag{18}$$

13 The VSM-inspired classifier for resolving CAs was mainly proposed and implemented by Wenfeng Yang when he was visiting Microsoft Research Asia.

Our experiments show that the VSM-inspired classifier outperforms other well-known classifiers for this particular task. For example, a maximum entropy classifier using the same features achieved an overall accuracy of 94.1%.

For the second problem of NE and FT detection, though, we can simply use the FSA-based approach, as described in Section 5.4, to detect FTs in the initial segmented corpus; our method of NER in the initial step (i.e., step 1) is a little more sophisticated. First, we manually annotate named entities on a small subset (called the seed set) of the training data. Then we obtain a context model on the seed set (called the seed model). We thus improve the context model that is trained on the initial annotated training corpus by interpolating it with the seed model. Finally, we use the improved context model in steps 2 and 3 of the bootstrapping. We shall show in the next subsection that a relatively small seed set (e.g., 150K words) is enough to get a reasonably good context model for initialization.

7.2 Evaluation Results

To justify the methods just described, we built a large number of context models using different initial corpora. For each of the initial corpora, a context model is trained using the Viterbi iterative procedure until convergence, i.e., the improvement of the word segmentation performance of the resulting system, is less than a preset threshold. The results are shown in Table 17, where Row 1 (FMM) presents the segmentation results of using the initial corpus segmented by a greedy word segmenter—the basic solution described earlier; in Row 2, we resolve segmentation (overlap) ambiguities on top of the corpus in Row 1; we then tag FTs in Rows 3 and 4. From Rows 5 to 8, several NE annotated seed sets of different sizes are used, showing the trade-off between performance and human cost. In Rows 1 to 8, we use the raw training set containing approximately 50 million characters. For comparison, we also include in Row 9 the results of MSRSeg, whose context model has been trained on a 20-million-word manually annotated corpus. The experimental results reveal several facts.

Table 17
Comparison of performance of MSRSeg: The versions that are trained using (semi-)supervised iterative training with different initial training sets (Rows 1 to 8) versus the version that is trained on annotated corpus of 20 million words (Row 9).

	Initial training set		Word segmentation			FT		PN		LN		ON	
			F	P	R	P	R	P	R	P	R	P	R
1	FMM		.877	.833	.927								
2	1 + OA		.886	.841	.936								
3	1 + FT		.919	.894	.946	.903	.887						
4	2 + FT		.927	.903	.954	.902	.886						
5	4 + NE	0.15	.961	.955	.968	.900	.889	.835	.818	.897	.865	.780	.681
6	Seed set	0.35	.967	.962	.973	.903	.891	.840	.820	.900	.866	.794	.680
7	(M word)	0.50	.967	.962	.972	.905	.895	.844	.821	.900	.863	.800	.678
8		1.00	.968	.963	.973	.905	.895	.854	.826	.903	.865	.802	.679
9	MSR training set		.974	.969	.979	.905	.899	.870	.906	.892	.855	.816	.654

Table 18
Precision of person name recognition on the MSR test set, using Viterbi iterative training, initialized by four seed sets with different sizes.

Seed set (M char)	w/o iteration	w/1 iteration	w/2 iterations
0.15	.745	.790 (+1.6%)	.800 (+0.7%)
0.35	.757	.796 (+1.8%)	.802 (+0.5%)
0.50	.783	.824 (+5.2%)	.835 (+1.3%)
1.00	.815	.835 (+2.5%)	.840 (+0.6%)

- Although the greedy segmenter (FMM) can resolve around 90% of ambiguities in word segmentation, as shown in Table 16, the resulting segmenter is still much worse than MSRSeg because a large number of unknown words cannot be detected correctly even after Viterbi iterative learning.
- The method of resolving OA brings marginal improvements. Since the method does not require any human annotation, Row 2 shows the best results we achieved in our experiments using unsupervised learning approaches.
- Factoid rules, although simple, bring substantial improvements.
- The Viterbi iterative training method does not turn out to be an effective way of resolving ambiguities in word segmentation or of detecting new words. In Rows 1 to 4, the word segmentation performance always saturates after 2 or 3 iterations, with little improvement. For example, FMM (Row 1) achieves an initial segmentation F-measure of 0.8771, and after two iterations, it saturates and ends up with 0.8773.

Table 19
Precision of location name recognition on the MSR test set, using Viterbi iterative training, initialized by four seed sets with different sizes.

Seed set (M char)	w/o iteration	w/1 iteration	w/2 iterations
0.15	.883	.895 (+1.4%)	.897 (+0.2%)
0.35	.893	.901 (+0.9%)	.900 (−0.1%)
0.50	.894	.901 (+0.8%)	.900 (−0.1%)
1.00	.895	.902 (+0.8%)	.903 (+0.1%)

Table 20
Precision of organization name recognition on the MSR test set, using Viterbi iterative training, initialized by four seed sets with different sizes.

Seed set (M char)	w/o iteration	w/1 iteration	w/2 iterations
0.15	.695	.770 (+10.8%)	.780 (+1.3%)
0.35	.737	.786 (+6.6%)	.794 (+1.0%)
0.50	.745	.790 (+6.0%)	.800 (+1.3%)
1.00	.757	.796 (+5.2%)	.802 (+0.8%)

- The Viterbi iterative training is effective in boosting the precision of NER without great sacrifices for recall (e.g., the recall remains almost the same when using the seed set of Row 5 in Table 17, or becomes a little worse when using the seed sets of Rows 6 to 8). As shown in Tables 18–20, we start with a series of seed sets of different sizes and achieve a reasonable accuracy of NER, which is comparable with that of MSRSeg, after two iterations.
- The use of a small NE annotated seed set (e.g., in Row 5) would achieve the best trade-off between performance and human effort, because after two iterations, the accuracy of NER is very close to that of using larger seed sets, while the human effort of creating the seed set is much less.

8. System Evaluation

8.1 System Results

Our system is designed so that components such as the FT detector and NE recognizer can be "switched on or off" so that we can investigate the relative contribution of each component to the overall word segmentation performance. To date, we have not done a separate evaluation of MDW recognition. We leave that to future work.

The main results are shown in Table 21. For comparison, we also include in the table (Row 1) the results of using the greedy segmenter (FMM) described in Section 7. Row 2 shows the baseline results of our system, where only the lexicon is used. It is interesting to find, in Rows 1 and 2, that the dictionary-based methods already achieve quite good recall, but the precision is not very good because those methods cannot correctly identify unknown words that are not in the lexicon, such as factoids and named entities. We also find that even using the same lexicon, our approach based on the linear mixture models outperforms the greedy approach (with a slight but statistically significant difference) because the use of context model resolves more ambiguities in segmentation. The most promising property of our approach is that the linear mixture models provide a flexible framework where a wide variety of linguistic knowledge and statistical models can be combined in a unified way. As shown in Rows 3 to 6, when components are switched on in turn by activating corresponding class models, the overall word segmentation performance increases consistently.

Table 21
MSRSeg system results for the MSR test set.

		Word segmentation		FT		PN		LN		ON	
	Segmenter	P	R	P	R	P	R	P	R	P	R
1	FMM	.837	.927								
2	Baseline	.863	.947								
3	2 + FT	.913	.961	.904	.898						
4	3 + PN	.950	.972	.905	.898	.790	.905				
5	4 + LN	.955	.975	.905	.899	.858	.906	.794	.860		
6	5 + ON	.969	.979	.905	.899	.870	.906	.892	.855	.816	.654

We also conduct an error analysis, which shows that 85% of errors come from NER and factoid detection, especially the NE abbreviations, although the tokens of these word types amount to only 8.3% in the MSR test set. The remaining 15% of errors are mainly due to new words.

8.2 Comparison with Other Systems using the MSR Test Set

We compare our system with three other Chinese word segmenters on the MSR test set:[14]

1. The MSWS system is one of the best available products. It is released by Microsoft (as a set of Windows APIs). MSWS first conducts word breaking using MM (augmented by heuristic rules for disambiguation), and then conducts factoid detection and NER using rules.
2. The LCWS system is one of the best research systems in mainland China. It is released by Beijing Language University. The system works similarly to MSWS, but has a larger dictionary containing more PNs and LNs.
3. The PBWS system is a rule-based Chinese parser (Wu and Jiang 2000) that can also output word segmentation results. It explores high-level linguistic knowledge such as syntactic structure for Chinese word segmentation and NER.

As mentioned earlier, to achieve a fair comparison, we compare the previously mentioned four systems only in terms of NER precision and recall and the number of OAS errors. However, we find that due to the different annotation specifications used by these systems, it is still very difficult to compare their results automatically. For example, 北京市政府 'Beijing city government' has been segmented inconsistently as 北京市/政府 'Beijing city' + 'government' or 北京/市政府 'Beijing' + 'city government' even in the same system. Worse still, some LNs tagged in one system are tagged as ONs in another system. Therefore, we have to manually check the results. We picked 933 sentences at random containing 22,833 words (including 329 PNs, 617 LNs, and 435 ONs) for testing. We also did not differentiate LNs and ONs in evaluation. That is, we only checked the word boundaries of LNs and ONs and treated both tags as interchangeable. The results are shown in Table 22. We can see that in this small test set, MSRSeg achieves the best overall performance of NER and the best performance of resolving OASs.

8.3 Evaluations on Bakeoff Test Sets

Table 23 presents the comparison results of MSRSeg on four Bakeoff test sets with others reported previously. The layout of the table follows (Peng, Feng, and McCallum 2004). SXX indicates participating sites in the 1st SIGHAN International Chinese Word Segmentation Bakeoff (Sproat and Emerson 2003). CRFs indicates the word segmenter reported in Peng, Feng, and McCallum (2004), which uses models of linear-chain con-

14 The three systems are well known in mainland China, but to our knowledge, no standard evaluations on Chinese word segmentation have been published. Although our comparison evaluations are limited to NER and crossing brackets (described later), we think the comparison we draw is still informative.

Table 22
Cross-system comparison results.

	# OAS	LN			PN			ON		
Segmenters	errors	P	R	F	P	R	F	P	R	F
MSWS	63	.935	.442	.600	.907	.744	.818	.642	.469	.600
LCWS	49	.854	.720	.782	.945	.781	.856	.713	.131	.222
PBWS	20	.767	.736	.752	.780	.787	.784	.817	.216	.342
MSRSeg	7	.876	.864	.870	.830	.897	.862	.799	.617	.696

Table 23
Comparisons against other segmenters: In Column 1, SXX indicates participating sites in the 1st SIGHAN International Chinese Word Segmentation Bakeoff, and CRFs indicates the word segmenter reported in (Peng et al. 2004). In Columns 2 to 5, entries contain the F-measure of each segmenter on different open runs, with the best performance in bold. Column Site-Avg is the average F-measure over the data sets on which a segmenter reported results of open runs, where a bolded entry indicates the segmenter outperforms MSRSeg. Column Our-Avg is the average F-measure of MSRSeg over the same data sets, where a bolded entry indicates that MSRSeg outperforms the other segmenter.

	ASo	ASc	CTBo	CTBc	HKo	HKc	PKo	PKc	Site-Avg	Our-Avg
S01			.881	.881		.901	.953	.951	.917	.930
S02			.912	.874					.912	.904
S03	.872		.829		.886		.925		.878	.943
S04							.937	.939	.937	.955
S05		.942		.732				.894		
S06		.945		.829		.924		.924		
S07							.940		.940	.955
S08					.956	.904	.938	.936	.947	.955
S09		.961						.946		
S10			.901	.831			.959	.947	.930	.930
S11	.904		.884		.879		.886		.888	.943
S12		.959				.916				
CRFs	.957	.956	.894	.849	.946	.928	.946	.941	.936	.943
MSRSeg	.958		.904		.954		.955			.943

ditional random fields (CRFs). Entries contain the F-measure of each segmenter on different open runs, indicated by XXo, with the best performance in bold. Column Site-Avg is the average F-measure over the data sets on which a segmenter reported results of open runs, where a bolded entry indicates the segmenter outperforms MSRSeg. Column Our-Avg is the average F-measure of MSRSeg over the same data sets, where a bolded entry indicates that MSRSeg outperforms the other segmenter. For completeness, we also include in Table 23 the results of closed runs, indicated by XXc. In a closed test, one can only use training material from the training data for the particular corpus being tested on. No other material was allowed (Sproat and Emerson 2003). Since MSRSeg uses the MSR corpus for training, our results are of open tests.[15]

15 It would be more informative if the comparison could be conducted in closed tests, which implies that dictionaries and models of MSRSeg should be generated solely on the given training data. We leave this for future work.

Several conclusions can be drawn from Table 23. First, for the same system, open tests generally achieve better results than closed tests due to the use of additional training material.[16] Second, there is no single segmenter that performs best in all four data sets. Third, MSRSeg achieves consistently high performance across all four data sets. For example, MSRSeg achieves better average performance than the other three segmenters that report results on all four data sets (i.e., S03, S11, CRF). In particular, MSRSeg outperforms them on every data set. There are two segmenters that achieve better average F-measure than ours. One is S02, which reported results on CTB only. The other is S10, which reported results on CTB and PK. From these results, we conclude that MSRSeg is an adaptive word segmenter that achieves state-of-the-art performance on different data sets, corresponding to different domains and standards.

As described in Section 2.1, most segmenters, including the ones in Table 23, can be roughly grouped into two categories: ones that use a rule-based approach and ones that use a statistical approach. MSRSeg is a hybrid system that takes advantage of both approaches. Though rule-based systems (e.g., S08, S10, and S11 in Table 23) can achieve reasonably good results, they cannot effectively make use of increasingly large training data and are weak in unknown word detection and adaptation. Some statistical segmenters (e.g., S01 and S07 in Table 23) use generative models such as HMM for Chinese word segmentation. However, it is very difficult to incorporate linguistic knowledge into the (approximated or assumed) generation process of Chinese sentences, underneath which the models are developed. Discriminative models (e.g., the linear models in MSRSeg, where though all components models are derived from generative models, they are combined using discriminatively trained weights) are free from this issue and provide a flexible mathematical framework to incorporate arbitrary linguistic knowledge. They do not assume any underlying generation process. Instead, they assume that the training and test sets are generated from the same distribution, but the form of the distribution (i.e., generative process) is unknown. If we view Chinese word segmentation as a classification problem, i.e., to discriminate between "good" segmentations and "bad" ones, we may prefer discriminative models to generative models. Intuitively, it is sufficient to find directly the desired features that can differentiate good segmentations from bad ones (as in discriminative models). It is, however, not necessary to estimate the distributions based upon which Chinese sentences are generated (or segmentations) first, and then use the estimated distributions to construct the desired features (as in generative models). As pointed out by Vapnik (1998): "When solving a given problem, solve it directly and try to avoid solving a more general problem as an intermediate step."

Our models are similar to the maximum entropy models in Xue (2003) and CRFs in Peng, Feng, and McCallum (2004) in that all these models give the flexibility to incorporate arbitrary features and can be discriminatively trained. Our models are novel in that many feature functions are derived from probabilistic or heuristic models inspired by source–channel models of Chinese sentence generation, as described in Section 4.3. Therefore, these feature functions are not only potentially more reasonable but also much more informative than, for instance, the binary features used in standard maximum entropy models in NLP.

16 It is interesting to see that, in AS, close tests achieve better performance than open tests, although among different systems. This is because the training data of AS is much larger than the other three corpora, and those segmenters that apply statistical approaches, such as S09 (Xue 2003) and CRFs, have been well trained.

We also notice that many segmenters (e.g., S03 and S04 in Table 23) separate unknown word detection from word segmentation. Though this would make the development of the segmenter easier, it seems to be a flawed solution in reality, as we discussed earlier. The benefits of integrating both tasks has also been shown empirically in Table 23.

9. Conclusions

This article presents a pragmatic approach to Chinese word segmentation. Our main contributions are threefold. First, we view Chinese words as *segmentation units* whose definition is pragmatic in nature and depends on how they are used and processed (differently) in realistic applications, while theoretical linguists define *words* using purely linguistic criteria. Second, we propose a pragmatic mathematical framework for Chinese word segmentation, where various problems of word segmentation (i.e., word breaking, morphological analysis, factoid detection, NER, and NWI) are solved in a unified approach. The approach is based on linear models where component models are inspired by source–channel models of Chinese sentence generation. Third, we describe in detail an adaptive Chinese word segmenter, MSRSeg. This pragmatic system consists of two components: (1) a generic segmenter that is based on the mathematical framework of word segmentation and unknown word detection, and that can adapt to different domain-specific vocabularies, and (2) a set of output adaptors for adapting the output of the former to different application-specific standards. Evaluation on five test sets with different standards shows that the adaptive system achieves state-of-the-art performance on all the test sets.

One area of our future work is to apply MSRSeg in a wide range of practical applications. We believe that some application-specific features can also be integrated into the framework. For instance, in MT, it would be interesting to investigate how to jointly optimize the performances of both word segmentation and word alignment.

As one of the reviewers pointed out, though the reliable high performance of MSRSeg is impressive, it is by far one of the most complex systems with access to the richest resources. Hence, another interesting area of our future work is to explore whether the performance is attributed to a superior architecture or simply to the richer resources. We have developed a simplified version of MSRSeg, called **S-MSRSeg.** It does not use the morph-lexicon and is trained using one-fifth of the MSR training data in Table 4, which are only partially labeled (i.e., LWs are not annotated). Interestingly, S-MSRSeg achieves very similar (or slightly worse) performance on the five test sets in Table 4. This demonstrates again the potential of our *pragmatic* approach to Chinese word segmentation. The work reported in this article represents not an end but a beginning of yet another view of Chinese word segmentation. Toward this end, S-MSRSeg and its training and test data sets are publicly available (e.g., at http://research.microsoft.com/∼jfgao) for the sake of encouraging others to improve upon the work we have carried out.

Acknowledgments

The work reported in this article was a team effort. The number of people intimately involved in this project is far too large for us to enumerate here. We only name a few now. We thank the manager teams of the NLC group at MSRA and the NLG at NISD, especially Ming Zhou, Hong-Jiang Zhang, Jun Liu, and Kai-Fu Lee, for initiating and funding the project. We thank Ashley Chang, Hongqiao Li, Jianfeng Li, Jianghong Li, Haowei Qin, Jian Sun, Xinsong Xia, Wenfeng Yang, Ye Zhang, and Xiaodan Zhu for their help in the experimentation and evaluation of our system, and for data annotation. We also thank John Chen, Nianwen Xue,

Richard Sproat, Keh Yi Su, Hisami Suzuki, and the three anonymous reviewers for their comments on early drafts of this article.

References

Aho, Alfred V., Ravi Sethi, and Jeffrey D. Ullman. 1986. *Compilers: Principles, Techniques, and Tools.* Addison-Wesley, Boston.

Baayen, R. Harald. 1989. *A Corpus-based Approach to Morphological Productivity: Statistical Analysis and Psycholinguistic Interpretation.* Ph.D. thesis, Free University, Amsterdam.

Berger, Adam, Stephen Della Pietra, and Vincent J. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71.

Brill, Eric. 1995. Transformation-based error-driven learning and natural language processing: A case study in Part-of-Speech tagging. *Computational Linguistics*, 21(4):543–565.

Chang, Eric, Jianlai Zhou, Yu Shi, and Chao Huang. 2001. Speech lab in a box: A Mandarin speech toolbox to Jumpstart speech related research. In *Eurospeech'2001*, pages 2799–2782, Aalborg, Denmark.

Chang, Jing-Shin and Keh-Yih Su. 1997. An unsupervised iterative method for Chinese new lexicon extraction. *International Journal of Computational Linguistics and Chinese Language Processing*, 2(2):97–148.

Chen, Aitao. 2003. Chinese word segmentation using minimal linguistic knowledge. In *The Second SIGHAN Workshop on Chinese Language Processing*. July 11–12, Sapporo, Japan.

Chen, Keh-Jiann and Ming-Hong Bai. 1998. Unknown word detection for Chinese by a corpus-based learning method. *International Journal of Computational Linguistics and Chinese Language Processing*, 3(1):27–44.

Chen, Stanley F. and Joshua Goodman. 1999. An empirical study of smoothing techniques for language modeling. *Computer Speech and Language*, 13:359–394.

Cheng, Kowk-Shing, Gilbert H. Yong, and Kam-Fai Wong. 1999. A study on word-based and integral-bit Chinese text compression algorithms. *JASIS*, 50(3):218–228.

Chien, Lee-Feng. 1997. PAT-tree-based keyword extraction for Chinese information retrieval. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM Press, New York.

Chu, Min, Hu Peng, Yong Zhao, Zhengyu Niu, and Eric Chang. 2003. Microsoft Mulan—a bilingual TTS system. In *Proceedings of the 2003 International Conference on Acoustics, Speech and Signal Processing*, vol. 1, pages 264–267.

Collins, Michael and Nigel Duffy. 2001. Convolution kernels for natural language. In *Advances in Neural Information Processing Systems (NIPS 14)*, pages 625–632, MIT Press.

Collins, Michael. 2002. Discriminative training methods for Hidden Markov Models: theory and experiments with the perceptron algorithm. In *Proceedings of the 2002 Conference on Empirical Method in Natural Language Processing*, pages 1–8.

Collins, Michael. 2004. Parameter estimation for statistical parsing models: theory and practice of distribution-free methods. Chapter 1 in Harry Bunt, John Carroll and Giorgio Satta, editors, *New Developments in Parsing Technology*, Kluwer.

Dai, Yubin, Christopher S. G. Khoo, and Tech Ee Loh. 1999. A new statistical formula for Chinese word segmentation incorporating contextual information. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 82–89.

Della Pietra, Stephen, Vincent Della Pietra, and John Lafferty. 1997. Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19, 380–393.

Duda, Richard O., Peter E. Hart, and David G. Stork. 2001. *Pattern Classification*. John Wiley & Sons, Inc.

Dudewicz, Edward J. and Satya N. Mishra. 1988. *Modern Mathematical Statistics*. Wiley Series in Probability and Mathematical Statistics, John Wiley & Sons, Inc.

Gao, Jianfeng and Kai-Fu Lee. 2000. Distribution based pruning of backoff language models. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, October 3–6, Hong Kong, pages 579–586.

Gao, Jianfeng, Joshua Goodman, and Jiangbo Miao. 2001. The use of clustering techniques for language modeling—application to Asian language. *Computational Linguistics and Chinese Language Processing*, 6(1):27–60.

Gao, Jianfeng, Joshua Goodman, Mingjing Li, and Kai-Fu Lee. 2002. Toward a unified approach to statistical language modeling for Chinese. *ACM Transactions on Asian Language Information Processing*, 1(1):3–33.

Gao, Jianfeng, Mu Li, and Chang-Ning Huang. 2003. Improved source-channel model for Chinese word segmentation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, July 7–12, Sapporo, Japan, pages 272–280.

Gao, Jianfeng, Andi Wu, Mu Li, Chang-Ning Huang, Hongqiao Li, Xinsong Xia, and Haowei Qin. 2004. Adaptive Chinese word segmentation. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, July 21–26, Barcelona, pages 462–469.

Gao, Jianfeng, Hao Yu, Peng Xu, and Wei Yuan. 2005. Minimum sample risk methods for language modeling. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*.

Frakes, William B. and Ricardo Baeza-Yates (eds.). 1992. *Information Retrieval*. Prentice Hall, Englewood Cliffs, NJ.

Fung, Pascale and Dekai Wu. 1994. Statistical augmentation of a Chinese machine-readable dictionary. In *Proceeding of the Second Annual Workshop on Very Large Corpora*, Kyoto, pages 180–181.

Katz, S. M. 1987. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE ASSP* 35(3):400–401.

Katz, S. M. 1996. Distribution of content words and phrases in text and language modeling. *Natural Language Engineering*, 1996(2):15–59.

Kneser, Reinhard and Hermann Ney. 1995. Improved backing-off for *m*-gram language modeling. In *Proceeding of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 181–184.

Krauth, W. and M. Mèzard. 1987. Learning algorithm with optimal stability in neural networks. *Journal of Physics A*, 20:745–752.

Hockenmaier, Julia and Chris Brew. 1998. Error driven segmentation of Chinese. In *Communications of COLIPS*, 8(1):69–84.

Huang, Chu-Ren, Keh-Jiann Chen, Chang Lili, and Feng-Yi Chen. 1997. Segmentation standard for Chinese natural language processing. *International Journal of Computational Linguistics and Chinese Language Processing*, 2(2):47–62.

Joachims, Thorsten. 2002. *Learning to Classify Text Using Support Vector Machines*. Kluwer.

Li, Hongqiao, Chang-Ning Huang, Jianfeng Gao, and Xiaozhong Fan. 2004. The use of SVM for Chinese new word identification. In *Proceedings of the First International Joint Conference on Natural Language Processing*, Sanya City, Hainan Island, China, March 22–24, pages 497–504.

Li, Mu, Jianfeng Gao, Chang-Ning Huang, and Jianfeng Li. 2003. Unsupervised training for overlapping ambiguity resolution in Chinese word segmentation. In *Proceedings of the Second SIGHAN Workshop on Chinese Language Processing*, July 11–12, Sapporo, Japan, pages 1–7.

Liang, Nanyuan. (梁南元) 1987. 书面汉语自动分词系统 – CDWS (A written Chinese automatic segmentation system). In 中文信息学报 (*Journal of Chinese Information Processing*), 2:44–52.

Lin, Ming-Yu, Tung-Hui Chiang, and Keh-Yi Su. 1993. A preliminary study on unknown word problem in Chinese word segmentation. In *Proceedings of the ROC Computational Linguistics Conference VI*, 119–141.

Manning, Christopher D. and Hinrich Schutze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA.

Mitchell, Tom M. 1997. *Machine Learning*. McGraw-Hill, New York.

Nie, Jian-Yun, Wanying Jin, and Mareie-Louise Hannan. 1994. A hybrid approach to unknown word detection and segmentation of Chinese. In *Proceedings of the International Conference on Chinese Computing*, pages 326–335, Singapore.

Och, Franz. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, July 7–12, Sapporo, Japan, pages 160–167.

Packard, Jerome. 2000. *The Morphology of Chinese: A Linguistics and Cognitive Approach*. Cambridge University Press, Cambridge.

Palmer, David D. 1997. A trainable rule-based algorithm for word segmentation. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, Madrid, pages 321–328.

Peng, Fuchun, Fangfang Feng, and Andrew McCallum. 2004. Chinese segmentation and new word detection using conditional random fields. In *COLING 2004*, pages 562–568.

Roche, Emmanuel and Yves Schabes. 1997. *Finite-State Language Processing*. MIT Press, Cambridge, MA.

Rosenfeld, Roland, Stanley F. Chen, and Xiaojin Zhu. 2001. Whole sentence exponential language models: A vehicle for linguistic statistical integration. *Computer Speech and Language*, 15(1):55–73.

Sproat, Richard and Tom Emerson. 2003. The first international Chinese word segmentation bakeoff. In *Proceedings of the Second SIGHAN Workshop on Chinese Language Processing*, July 11–12, Sapporo, Japan.

Sproat, Richard and Chilin Shih. 2002. Corpus-based methods in Chinese morphology and phonology. In *Proceedings of the 2002 International Conference on Computational Linguistics*.

Sproat, Richard, Alan W. Black, Stanley Chen, Shankar Kumar, Mari Ostendorf, and Christoper Richards. 2001. Normalization of non-standard words. *Computer Speech and Languages*, 15:287–333.

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

Sproat, Richard. 1992. *Morphology and Computation*. MIT Press, Cambridge, MA.

Sproat, Richard and Chilin Shih. 1990. A statistical method for finding word boundaries in Chinese text. *Computer Processing of Chinese and Oriental Languages*, 4:336–351.

Su, Keh-Yih and Chin-Hui Lee. 1994. Speech recognition using weighted HMM and subspace projection approaches. *IEEE Trans., Speech and Audio Processing*, 2(1):69–79.

Sun, Jian, Jianfeng Gao, Lei Zhang, Ming Zhou, and Chang-Ning Huang. 2002. Chinese named entity identification using class-based language model. In *Proceedings of the 2002 International Conference on Computational Linguistics*, pages 967–973.

Sun, Jian, Ming Zhou, and Jianfeng Gao. 2003. Chinese named entity identification using class-based language model. *International Journal of Computational Linguistics and Chinese Language Processing*, 8(2):1–28.

Sun, Maosong. (孙茂松) and Benjamin K. Tsou (邹嘉彦). 2001. 汉语自动分词研究评述 (A review and evaluation on automatic segmentation of Chinese). 当代语言学 (Contemporary Linguistics), 3(1):22–32.

Sun, Maosong, Dayang Shen, and Benjamin K. Tsou. 1998. Chinese word segmentation without using lexicon and hand-crafted training data. In *Proceedings of the 17th International Conference on Computational Linguistics*, Montreal, Quebec, Canada, pages 1265–1271.

Sun, Maosong and Zhengping Zuo. 1998. Overlapping ambiguity in Chinese text. (In Chinese) *Quantitative and Computational Studies on the Chinese Language*, HK, pages 323–338.

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

Uchimoto, Kiyotaka, Satoshi Sekine, and Hitoshi Isahara. 2001. The unknown word problem: a morphological analysis of Japanese using maximum entropy aided by a dictionary. In *Proceedings of the 2001 Conference on EMNLP*, pages 91–99.

Vapnik, Vladimir N. 1998. *Statistical Learning Theory*. John Wiley & Sons, Inc.

Wu, Andi and Zixin Jiang. 2000. Statistically-enhanced new word identification in a rule-based Chinese system. In *Proceedings of the Second ACL Chinese Processing Workshop*, pages 41–66.

Wu, Andi. 2003. Customizable segmentation of morphologically derived words in Chinese. In *International Journal of Computational Linguistics and Chinese Language Processing*, 8(1):1–27.

Wu, Zimin and Gwyneth Tseng. 1993. Chinese text segmentation for text retrieval achievements and problems. *JASIS*, 44(9): 532–542.

Xia, Fei. 1999. *Segmentation guideline, Chinese Treebank Project*. Technical report, University of Pennsylvania.

Xue, Nianwen. 2003. Chinese word segmentation as character tagging. *International Journal of Computational Linguistics and Chinese Language Processing*, 8(1):29–48.

Zhu, Xiaodan, Mu Li, Jianfeng Gao, and Chang-Ning Huang. Single character Chinese named entity recognition. In *Proceedings of the Second SIGHAN Workshop on Chinese Language Proceeding*, July 11–12, Sapporo, Japan, pages 125–132.