Effectiveness of Methods for Syntactic and Semantic Recognition of Numeral Strings: Tradeoffs Between Number of Features and Length of Word N-Grams

Kyongho Min[1], William H. Wilson[1], and Byeong-Ho Kang[2]

[1] School of Computer Science and Engineering
University of New South Wales, Sydney, Australia
{min, billw}@cse.unsw.edu.au
[2] School of Computing
University of Tasmania, Hobart, Australia
bhkang@utas.edu.au

Abstract. This paper describes and compares the use of methods based on N-grams (specifically trigrams and pentagrams), together with five features, to recognise the syntactic and semantic categories of numeral strings representing money, number, date, etc., in texts. The system employs three interpretation processes: word N-grams construction with a tokeniser; rule-based processing of numeral strings; and N-gram-based classification. We extracted numeral strings from 1,111 online newspaper articles. For numeral strings interpretation, we chose 112 (10%) of 1,111 articles to provide unseen test data (1,278 numeral strings), and used the remaining 999 articles to provide 11,525 numeral strings for use in extracting N-gram-based constraints to disambiguate meanings of the numeral strings. The word trigrams method resulted in 83.8% precision, 81.2% recall ratio, and 82.5% in F-measurement ratio. The word pentagrams method resulted in 86.6% precision, 82.9% recall ratio, and 84.7% in F-measurement ratio.

Keywords: numeral strings, N-grams, named entity recognition, natural language processing.

1 Introduction

Numbers (e.g. “801 voters”) and Alpha-numeric strings (e.g. “9:30am”) with or without Roman numerals (e.g. “ENIAC-II”) and with or without special symbols (e.g. “$” in “$2.5 million”) are essential components of written or spoken human languages. Without understanding numeral strings, it is often hard to understand the sentences and texts that they appear in. Dale [6] discussed types of numerical expressions and their corresponding meanings in his 1997 paper on tokenisation of text: numeral strings are on average much more ambiguous than words. For example, the number “2” could mean AGE, RANK, ORDER, Section number, QUANT, DAY, NUMBER, part of a Named Entity, and etc.

M.A. Orgun and J. Thornton (Eds.): AI 2007, LNAI 4830, pp. 445–455, 2007.

Current NLP systems treat such strings as either a numeral (e.g. "801 voters") or as a named entity (e.g. MONEY for "$2.5 million"). However, ambiguity in their interpretation may arise without semantic/contextual information: for example, "28" in the phrase "he turns 28 today" can on the surface be interpreted as any of the following - (a) as NUMBER; (b) as DAY of a date expression; or (c) as AGE at the lexical meaning level. Such numeral strings are called *separate numeral strings* in this paper. There are also *affixed numeral strings* (e.g. "*10m* yacht"); these have meaningful semantic units attached (e.g. "m" in "10m") that reduce interpretation ambiguity. The natural interpretation in this case is LENGTH, but note that "10m" could signify a model number, or, if embedded in "$10m yacht", MONEY. In another context, "m" could signify "minute", so that "10m" would be TIME.

A special method related to N-grams, called s-grams (Skipped grams), was studied in [8] in the context of approximate string matching in an Information Retrieval area for European languages. The complex use of numeral strings in biomedical texts for protein name identification has been studied [19] [21]. In the domain of text classification, meanings of numeral strings (e.g. Time, Date, Money, Phone number etc.) based on bigrams were used as contextual features in call-for-tender documents [16]. Agrawal and Srikant discussed a keyword searching method for {attribute, numeric_value} pairs in the database (e.g. a query like "address set-up speed 20 ns power 500mW CMOS") [1]. Even though their task is not directly related to our task, this is an instance of the importance of meanings of numerals in information retrieval areas.

In the information extraction and named entity recognition areas [3], [5], basic semantic categories (e.g. PERSON, ORGANISATION, DATE, TIME, and MONEY) have been recognised. FACILE [3] in MUC used a rule-based named entity recognition system incorporating a chart-parsing technique with semantic categories. Semantic categories and semantic tags were used for a Chinese classification system in [12], [22] and for Japanese documents in [2], [20]. The ICE-GB grammar [15] used *cardinal*, *ordinal*, *fraction*, *hyphenated*, or *multiplier,* with two number features - singular and plural – for numeral categories. Zhou and Su [23] employed an HMM-based chunk tagger to recognise numerical quantities with surface and semantic features like FourDigitNum (e.g. 1990) as a year form, and SuffixTime (e.g. a.m.) as a time suffix (see also [18] for time phrases in weather forecasts). Polanyi and van den Berg [17] studied anaphoric resolution of quantifiers and cardinals and employed a quantifier logic framework. Min et. al. [13], [14] studied a manually generated rule-based numeral-string interpretation method and also a system based on automatically-generated bigrams constraints.

We have now implemented ENUMS (English NUMeral understanding System) with trigrams and also with 5-grams (e.g. 3L1R1 – a trigram consisting of one token to the left of the numeral string and one to the right, and 5L2R2) using constraints based on five features: POS, syntactic features, prefix, suffix, and special information. The ENUMS system is composed of a tokeniser, word N-grams retrieval, constraints retrieval, and numeral strings recognition and disambiguation modules in detail, with four sources of knowledge bases: dictionary, morphological rules, syntactic rules of a numeral strings, and word N-gram constraints. The understanding of numeral strings depends on their type: affixed numeral strings (e.g. "24km/h") require rule-based processing (i.e. syntactic rules for affixed numeral strings) after deep tokenisation of the numeral string (e.g. "24km/h" → "24" + "km" + "/" + "h"). In this paper, we will

focus on the tradeoffs between size of N-grams (e.g. 3 in 3L1R1 vs 5 in 5L2R2) and number of features used for constraint retrieval (in the range 1 to 5).

In the next section, the architecture of the ENUMS system will be described. In section 3, we will describe the ENUMS recognition algorithm. Section 4 will describe experimental results obtained with trigrams and pentagrams variants of ENUMS, and discussion and conclusions follow.

2 Architecture of the ENUMS System

An ENUMS system is organsised into modules using different knowledge sources: a dictionary, morphological rules for words, syntactic rules for numeral strings, word-N-gram-based constraints using a number of features (e.g. POS, syntactic features, prefix, suffix). ENUMS is implemented in Common Lisp with its IDE. In the next section, we will focus on the recognition of numeral strings using ENUMS.

We discuss categories and rules used in the ENUMS system briefly in this section. Table 1 shows some examples of numeral strings, and syntactic and semantic categories used in the ENUMS system. We used 40 numeral string categories – some are of a semantic nature, such as MONEY and DATE, and some are basically syntactic, such as NUMBER, FLOATNUMBER, for numeral strings with no more specific category. The category FMNUMBER (ForMatted Number) signifies numbers that include commas every 3 digits to the left of the unit digit for ease of reading, as in "*12,000* peacekeepers."

Table 1. Some categories and examples of them

Category	Example	Category	Example
Day	"August *11*"	Age	"mature *20-year-old* contender
Floatnumber	"support at *26.8* per cent"	Daytime	"between *9:30am* and *2am*"
Number	"*8000* of the Asian plants"	Length	"a *10m* yacht"
Quant	"survey of *801* voters"	Money	"spend *US$1.4* billion"
Year	"by September *2026*"	Scores	"a narrow *3-6* away loss to Otago"

The ENUMS dictionary includes meaningful objects useful for interpretation, such as symbolic tokens (e.g. "(", ")"), lexical words (e.g. "year"), and units (e.g. "km", "m", etc.). For example, the lexical information for "m" is ("m" ((:POS TU :SEM MINUTE) (:POS LU :SEM METER) (:POS NU :SEM MILLION))) where TU stands for Time Unit, LU Length Unit, NU Number Unit, and SEM SEMantics.

Fig. 1 describes the architecture of the ENUMS system. There are four modules with four major types of knowledge bases. The Tokeniser module reads input documents and analyses each document into sentences and tokens based on a word (i.e. a string delimited on the left and right by spaces). This module creates an object called DOCUMENT including information such as the boundary position of each sentence and the tokens in each sentence, with their positional information. Each token has much information, including, importantly, the type of token: WORD, NUMERAL, or SYMBOL.

The token classification is based on the following simple rule:

1) If a string includes any digit, then it is a NUMERAL string;
2) If a string only includes alphabets, then it is a WORD; or
3) Otherwise, it is a SYMBOL.

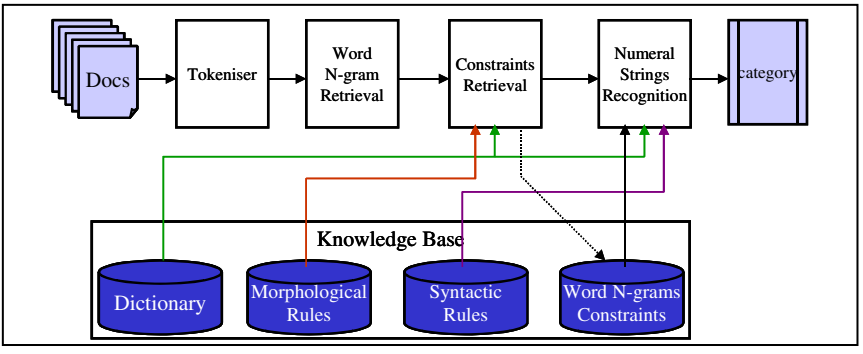

Fig. 1. Architecture of an ENUMS system

For the purpose of sentence boundary understanding, the Tokeniser makes two dummy tokens: BOS – Beginning Of Sentence - and EOS – End Of Sentence. When retrieving the word N-grams for a numeral string, BOS or EOS is the leftmost/rightmost token of word N-grams.

Table 2 shows an example of N-grams constraints. The constraint with ID 3 would hold for a word that is one position (:OFFSET 1) to the right (:D R) of the numeral string, if the word was an adverb (:POS ADV) and it was in capitals (:SPEC CAPITAL).

Table 2. Examples of N-grams constraints based on five features

ID	:D	:OFFSET	:POS	:SYNF	:PREFIX	:SUFFIX	:SPEC
1	L	1	NOUN	PLUR	NIL	NIL	NIL
2	R	1	PREP	“by”	NIL	NIL	CAPITAL
3	R	1	ADV	NIL	NIL	NIL	CAPITAL
4	L	1	PREP	“to”	NIL	NIL	NIL
5	R	1	PREP	“per”	NIL	NIL	PERCENT

The Word N-gram Retrieval module retrieves N consecutive words surrounding each numeral string, including the numeral string itself. The N-gram is described using a NL_iR_j format, signifying N words, i of them to the left and j to the right of the numeral string – so N = i + j – 1. So 3L1R1 means a trigram including the numeral string and one word to its left and right. Similarly, 4L1R2 signifies a tetragram with one word to the left and two to the right of the numeral string. We say that an N-gram of format NL_iR_j is *balanced* if i = j, otherwise it is *unbalanced*. In this paper, we focus on balanced N-grams of formats 3L1R1 and 5L2R2.

The third module is a Constraints Retrieval module. This module retrieves constraints relating to the numeral string's word N-grams with features of words in the N-grams. Each constraint is composed of one to five features as shown in Table 2: POS, syntactic feature, prefix of a numeral string (e.g. the "(" in the numeral string "(02)" that can signify a long distance prefix in a phone number), suffix of a numeral string, and special features (e.g. CAPITAL in "USD" and "GBP", some semantic concepts like MONTH for "January").

Table 3. Examples of category frequency of N-gram constraints (for 3L1R1)

ID	Category Frequency of N-grams Constraints
(1 2)	((MONEY 2) (RATE 1) (AGE 1) (QUANT 1) (NUMBER 1) (DATE 17))
(1 3)	((DATE 51))
(1 5)	((SCORES 3) (CENT 3) (AGE 2) (NUMBER 3) (YEAR 8) (QUANT 3) (DATE 55))
(4 1)	((SCORES 3) (YEAR 1) (AGE 2) (NUMBER 2) (PLURAL 1) (DAY 5) (QUANT 4))
(4 2)	((RANGE 1) (FLOATNUMBER 32) (QUANT 67))

Each constraint will be associated with a particular category frequency distribution in the training data, and this is illustrated in Table 3. Note that actual IDs used are pairs of numbers, like (1 3). It can be seen that constraint (1 3) is always associated with the category DATE in the data, whereas (1 5) is mostly DATE, but also has a scattering of other outcomes. The categories of the numeral strings in the training data (and the test data) were obtained by hand-annotation.

3 ENUMS Recognition Algorithm

In this section, the algorithm of the ENUMS system is discussed, based on word N-grams and associated constraints extracted from training data using up to five features. When a numeral string is found, its N-gram is determined, and this information is passed to the recognition module. The recognition algorithm is described by the pseudocode in Fig. 2.

With a numeral string that contains an affix, the string is also processed by deep tokenisation (e.g. "20.08.2003" → "20 + "." + "08" + "." + "2003"). Then a chart parsing technique [7] using 91 context-free rules that represent the structural form of affixed numeral strings is employed. Each rule is composed of a LHS (left hand side), RHS (right hand side), and constraints on the RHS. For example,

Rule7	RULE-ID:	R7
	LHS:	DATE
	RHS:	(DAY DOT MONTH DOT YEAR)
	Constraints:	((LEAPDATEP DAY MONTH YEAR))

If the recognition module is unable to choose between two or more categories, then word N-gram constraints extracted from the training data are used to select the best single category. For example, if the test numeral string "91 rules", whose annotated category is QUANT, generated word trigrams constraints like (1 2) and (4 2) in Table 3, then the simple addition of categorical frequency of both constraints

would be (MONEY 2) (RATE 1) (AGE 1) (QUANT 68) (NUMBER 1) (DATE 17) (RANGE 1) (FLOATNUMBER 32). The most frequent category is QUANT and this would be the disambiguated category for the test numeral string. We employed simple addition of categorical frequency obtained from training data and we plan to implement complex similarity (e.g. Cosine similarity), classification, probability (Bayes rule), or machine learning methods later, and compare these with our current results.

Recognise a numeral string's syntactic/semantic category with the following knowledge bases: dictionary, morphological rules, syntactic rules, and word N-gram constraints.

Input definition: Numeral strings with their word N-grams.
with detected type of a numeral string: affixed (e.g. "20km/h"), or separate ("2007"),
IF numeral string type = affixed
THEN
Apply numeral string tokeniser (e.g. "20" + "km" + "/" + "h")
Apply a chart parsing technique to the tokenised strings with rules for numeral string processing (e.g. syntactic rules)
IF Resulting category is NOT ambiguous,
THEN return the resulting syntactic/semantic category,
ELSE disambiguate the categories by using word N-gram constraints (i.e. the most frequent category is chosen using word N-gram constraints collected from train data),
END IF

ELSE (numeral string type = separate)
Apply separate numeral string understanding rules (which will suggest categories like AGE, NUMBER, DAY, etc.)
IF Resulting category is NOT ambiguous,
THEN return the resulting syntactic/semantic category,
ELSE disambiguate the categories by using word N-gram constraints (e.g. the most frequent category is chosen using word N-gram constraints collected from train data).
END IF
END IF

Fig. 2. Recognition algorithm for numeral strings

4 Experimental Results

We collected online newspaper articles (1,111 articles) for a month that cover a range of topics such as domestic, international, sports, economy, and etc. The articles included 12,803 (2.2%) numeral strings among a total of 594,588 strings. To test this system, we produced 10 sets of training and test data. For each set, we randomly selected 10% of test documents (112 among 1,111 documents). Documents in each test set did not overlap each other. Then, for each document set, testing was done using constraints derived using 1, 2, 3, 4, or all 5 features – that is, with just :POS, with :POS and :SYNF, with :POS, :SYNF, and :PREFIX, with :POS, :SYNF, :PREFIX, and :SUFFIX, and with all 5 features. Thus there were in all 10×5 = 50 test setups.

Table 4. Average data size of N-gram approaches

Date Name	Total Articles		Total Numerals		Total Strings	
	Train	Test	Train	Test	Train	Test
Rule-based (average)	91	287 (41)	886	3,215 (459)	48,498	144,030 (20,576)
2L1R0/2L0R1	83	295	915	3,222	* 192,528	
3L1R1	999	111	11,525	1,278	534,525	60,063
5L2R2	999	111	11,525	1,278	534,525	60,063

(* the training and test data were not computed separately.)

Table 4 shows the data size of articles, total numeral strings, and total strings used for training and testing the ENUMS system. The proportion of numeral strings belonging to each category based on 1,111 articles were QUANT (2,245 of 12,803, 17.5%, e.g. "survey of *801* voters"), MONEY (1,282, 10.0%, e.g. "$15m", "$2.55"), DATE (1,114, 98.7%, e.g. "02.12.2003"), FLOATNUMBER (1,099, 8.6%, e.g. "12.5 per cent"), YEAR (1,066, 8.3%, e.g. "in 2003"), NUMBER (1,056, 8.2%, e.g. "800 of the Asian plants"), and DAYTIME (704, 5.5%, e.g. "at 2:30pm") in order. In this paper, the size of total data was tripled compared to [14].

Table 5. Recall/Precision/F-measurement ratios of word N-grams with five features

Test Methods	Precision ratio (%)	Recall ratio (%)	F-measure (%)
Feat-1-3L1R1 Method (Average)	84.4	76.3	80.1
Feat-2-3L1R1 Method (Average)	84.0	79.4	81.6
Feat-3-3L1R1 Method (Average)	83.1	79.5	81.2
Feat-4-3L1R1 Method (Average)	83.7	80.7	82.2
Feat-5-3L1R1 Method (Average)	83.8	81.2	82.5
Feat-1-5L2R2 Method (Average)	85.5	80.8	83.1
Feat-2-5L2R2 Method (Average)	86.2	82.3	84.2
Feat-3-5L2R2 Method (Average)	86.2	82.3	84.2
Feat-4-5L2R2 Method (Average)	86.4	82.7	84.5
Feat-5-5L2R2 Method (Average)	86.6	82.9	84.7

Table 5 shows the performance using word trigrams (i.e. 3L1R1) and pentagrams (i.e. 5L2R2) with five features. The performance depended on the number of features and the number of N-grams. The performance of Feat-1-3L1R1 is the worst among five 3-L1R1 methods and the performance of Feat-1-5L2R2 is the worst among five 5L2R2 methods. The interesting thing in our test is that the number of features did not guarantee improved performance. In the word trigrams, Feat-2 to Feat-5 showed worse performance in precision ratio than Feat-1 (i.e. the differences are 0.4%, 1.3%, 0.7%, and 0.6% respectively).

In the case of the word pentagrams, the performance based on number of features improved monotonically. The addition of :SYNF information (i.e. going from Feat-1-5L2R2 to Feat-2-5L2R2) improved the F-measure by 1.1%. However, the addition of prefix information of a numeral string (i.e. Feat-3-5L2R2) did not improve (or decrease) F-measure. Fig. 3 to Fig. 8 shows the performance of 10 tests with five

feature sets in terms of precision, recall, and F-measurement ratios. The performance greatly depended on source knowledge of training data.

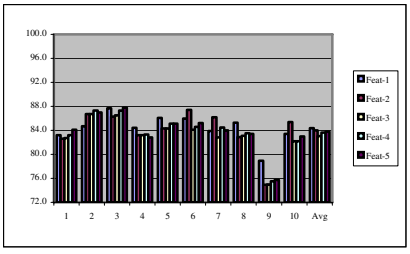

Fig. 3. Precision ratios (%) of 10 tests based on a 3L1R1 method

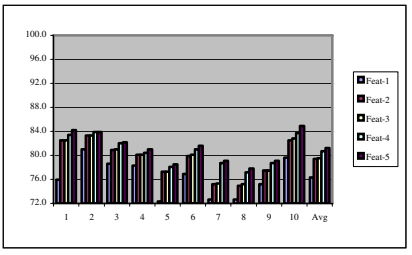

Fig. 4. Recall ratios (%) of 10 tests based on a 3L1R1 method

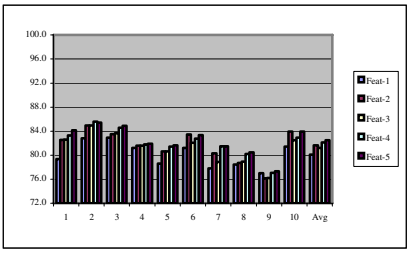

Fig. 5. F-measurement ratios (%) of 10 tests based on a 3L1R1 method

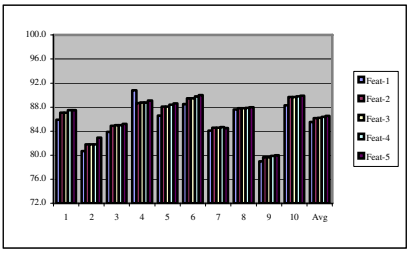

Fig. 6. Precision ratios (%) of 10 tests based on a 5L2R2 method

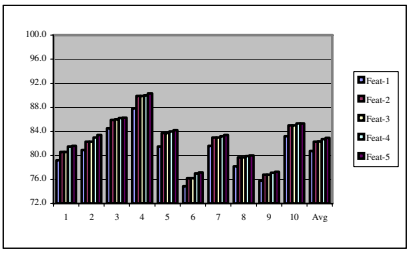

Fig. 7. Recall ratios (%) of 10 tests based on a 5L2R2 method

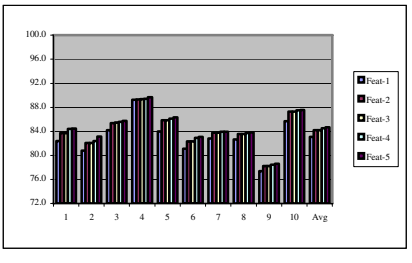

Fig. 8. F-measurement ratios (%) of 10 tests based on a 5L2R2 method

Table 6 shows the performance comparison between rule-based methods [13] and bigrams [14] and our current results with trigrams and pentagrams. The system in [13] was based on a manually generated rule-based method and the system in [14] was based on an automatically generated tabular feature-based method based on two types of bigrams (2L1R0 and 2L0R1).

Table 6. Performance comparison between methods with five features

Test Methods	Precision ratio (%)	Recall ratio (%)	F-measure (%)
Rule-based Method [13]	**86.8**	77.1	81.6
Bigrams Method (Average) [14]	74.5	68.1	71.2
Bigrams Method (best) [14]	83.1	74.5	78.6
Bigrams Method (worst) [14]	57.2	61.1	59.1
Trigrams Method with five features	83.8	81.2	82.5
Pentagrams Method with five features	86.6	**82.9**	**84.7**

The current systems were based on word trigrams (3L1R1), and word pentagrams (5L2R2). The rule-based method did best in precision ratio (86.6%) though the difference compared with pentagrams is just 0.2%. On Recall and F-measurement ratios, pentagrams performed better than other methods. Conclusively, the performance of the time-consuming and manually obtained rule-based method is not better than that of the easy automatic pentagrams method and the performance of the rule-based method greatly depends on the number of rules encoded.

5 Discussion and Conclusions

We focused on interpretation of varieties of numeral strings. Thus direct comparison of our system with other Named Entity recognition systems is not very meaningful, because the systems in e.g. MUC-7 [3] and CoNLL2003 [5] focused on the general recognition task for named entities, using a more limited set of categories: person, location, date, money, and organization, whereas we used 40 categories. In addition, the systems in MUC-7 were trained and tuned by using a training corpus with document preprocessing (e.g. tagging and machine learning). However, performances of MUC-7 systems ranged from 73% to 97% in precision, while our system correctly interpreted 86.6% of the numeral strings using the larger set of categories of numeral string.

For further improvement, different methods related to classification, similarity, and probability will be considered (e.g. application of cosine similarity, Bayes rules, decision tree, etc.). Secondly, the representation of knowledge (i.e. N-gram constraints) will be considered in order to adapt the system to knowledge acquisition (KA) methods like MCRDR (Multiple Classification Ripple-Down Rules) [9], [10]. Third, the integrated method of KA and machine learning algorithms could be applied to extracting domain knowledge (i.e. optimised N-grams constraints) and its model from training data [11]. In this paper, no optimising or learning techniques were applied to get best/optimised N-grams constraints. Fourth, the best value for N in word N-grams (the present system can handle 2 to Max where Max is the maximum string length of the longest sentence) and more effective features will be studied. In addition, the extension of this system to other data like standard biomedical corpora [19] [21] will help to test the effectiveness of our approach.

In conclusion, separate and affixed numeral strings are frequently used in real text. However, there is no system that interprets numeral strings systematically; they are frequently treated as either numerals or nominal entities. In this paper, we discussed two N-gram methods and compared their performance with a manually obtained rule-based method [13] and a word bigrams method [14]. The word trigrams method with

five features performs better than rule-based and word bigrams methods in terms of the F-measurement ratio. The word pentagrams method with five features was 0.2% behind in precision ratio but 2.9% ahead in F-measurement ratio compared with the rule-based method. We also found that the addition of extra features to obtain N-gram constraints did not always improve the recognition performance.

References

1. Agrawal, R., Srikant, R.: Searching with Numbers. In: Proceedings of WWW2002, pp. 190–196 (2002)
2. Asahara, M., Matsumoto, Y.: Japanese Named Entity Extraction with Redundant Morphological Analysis. In: Proceedings of HLT-NAACL 2003, pp. 8–15 (2003)
3. Black, W., Rinaldi, F., Mowatt, D.: Description of the NE system used for MUC-7. In: Proceedings of MUC-7 (1998)
4. Chieu, L., Ng, T.: Named Entity Recognition: A Maximum Entropy Approach Using Global Information. In: Proceedings of the 19th COLING, pp. 190–196 (2002)
5. CoNLL-2003 Language-Independent Named Entity Recognition. http://www.cnts.uia.ac.be/conll2003/ner/2 (2003)
6. Dale, R.: A Framework for Complex Tokenisation and its Application to Newspaper Text. In: Proceedings of the second Australian Document Computing Symposium (1997)
7. Earley, J.: An Efficient Context-Free Parsing Algorithm. CACM 13(2), 94–102 (1970)
8. Jarvelin, A., Jarvelin, A., Jarvelin, K.: s-grams: Defining Generalised n-grams for Information Retrieval. Information Processing and Management. 43, 1005–1019 (2007)
9. Kim, Y., Park, S., Kang, B., Choi, Y.: Incremental Knowledge Management of Web Community Groups on Web Portals. In: Karagiannis, D., Reimer, U. (eds.) PAKM 2004. LNCS (LNAI), vol. 3336, pp. 198–207. Springer, Heidelberg (2004)
10. Kim, Y., Park, S., Kang, B., Deards, E.: Adaptive Web Document Classification with MCRDR. In: ITCC 2004. Proceedings of International Conference on Information Technology, Las Vegas USA, pp. 198–207 (2004)
11. Mahidadia, A., Compton, P.: Knowledge Management in Data and Knowledge Intensive Environments. In: Karagiannis, D., Reimer, U. (eds.) PAKM 2004. LNCS (LNAI), vol. 3336, pp. 10–116. Springer, Heidelberg (2004)
12. Maynard, D., Tablan, V., Ursu, C., Cunningham, H., Wilks, Y.: Named Entity Recognition from Diverse Text Types. In: Proceedings of Recent Advances in NLP (2001)
13. Min, K., Wilson, W.H., Moon, Y.: Syntactic and Semantic Disambiguation of Numeral Strings Using an N-gram Method. In: Zhang, S., Jarvis, R. (eds.) AI 2005. LNCS (LNAI), vol. 3809, pp. 82–91. Springer, Heidelberg (2005)
14. Min, K., Wilson, W.H.: Comparison of Numeral Strings Interpretation: Rule-base and Feature-Based N-gram Methods. In: Sattar, A., Kang, B.H. (eds.) AI 2006. LNCS (LNAI), vol. 4304, pp. 1226–1230. Springer, Heidelberg (2006)
15. Nelson, G., Wallis, S., Aarts, B.: Exploring Natural Language - Working with the British Component of the International Corpus of English, John Benjamins, Netherlands (2002)
16. Paradis, F., Nie, J.: Contextual Feature Selection for Text Classification. Information Processing and Management. 43, 344–352 (2007)
17. Polanyi, L., van den Berg, M.: Logical Structure and Discourse Anaphora Resolution. In: ACL 1999. Proceedings of Workshop on The Relation of Discourse/Dialogue Structure and Reference, pp. 110–117 (1999)

18. Reiter, E., Sripada, S.: Learning the Meaning and Usage of Time Phrases from a parallel Text-Data Corpus. In: HLT-NAACL2003. Proceedings of Workshop on Learning Word Meaning from Non-Linguistic Data, pp. 78–85 (2003)
19. Seki, K., Mostafa, J.: A Hybrid Approach to Protein Name Identification in Biomedical Texts. Information Processing and Management. 41, 723–743 (2005)
20. Siegel, M., Bender, E.M.: Efficient Deep Processing of Japanese. In: Proceedings of the 3rd Workshop on Asian Language Resources and International Standardization (2002)
21. Torii, M., Kamboj, S., Vijay-Shanker, K.: An investigation of Various Information Sources for Classifying Biological Names. In: Dignum, F.P.M. (ed.) ACL 2003. LNCS (LNAI), vol. 2922, pp. 113–120. Springer, Heidelberg (2004)
22. Wang, H., Yu, S.: The Semantic Knowledge-base of Contemporary Chinese and its Apllication in WSD. In: Proceedings of the Second SIGHAN Workshop on Chinese Language Processing, pp. 112–118 (2003)
23. Zhou, G., Su, J.: Named Entity Recognition using an HMM-based Chunk Tagger. In: Proceedings of ACL2002, pp. 473–480 (2002)