A Two-Stage Incremental Annotation Approach to Constructing A Network Informal Language Corpus

Yunqing Xia   Kam-Fai Wong
Department of S.E.E.M.
The Chinese University of Hong Kong
Shatin, Hong Kong
{yqxia,kfwong}@se.cuhk.edu.hk

Robert Luk
Department of Computing
Hong Kong Polytechnic University
Hong Hum, Hong Kong
csrluk@comp.polyu.edu.hk

Abstract

Network Informal Language (NIL) refers to the special human language widely used in the community of digital network chat via platforms such as chat rooms/tools, mobile phone short message services (SMS), bulletin board systems (BBS), emails, etc. NIL holds anomalous characteristics in forming words, phrases, and non-alphabetical characters. This makes it difficult to handle NIL text by conventional natural language processing (NLP) tools. Previous research reveals that knowledge based methods perform less effectively in processing unseen NIL expressions. This motivates us to construct an annotated NIL corpus which is used specially to develop and evaluate techniques for extraction and normalization of NIL expressions. A two-stage incremental annotation approach is proposed in this paper to construct a NIL corpus with minimal human involvement. Several experiments are conducted which reveal that the efficiency of corpus annotation can be improved greatly with this approach.

Keywords: *network informal language, corpus annotation, natural language processing*

1 Introduction

Network Informal Language (NIL) refers to the special human language widely used in the community of network chat via platforms such as chat rooms/tools, mobile phone short message services (SMS), bulletin board systems (BBS), emails, etc. NIL is ubiquitous due in special to the rapid proliferation of Internet applications. NIL appears frequently within increasing volume of chat logs of online education [1] and customer relationship management [2] via chat rooms/tools. In wed-based chat rooms and BBS a large volume of NIL text is abused by solicitors of terrorism [3], pornography and crime [4]. A survey by the Global System for Mobile Communication (GSM) showed that Germans send 200 million messages a year [5]. All the facts disclose the growing importance in processing NIL text. This is a brand new research topic in NLP research area and its significance to NLP applications is obvious. To the best of our knowledge, no reported work has been published in this area.

NIL holds anomalous characteristics in forming non-alphabetical characters, words, and phrases. For example, "b4" is used to replace "before" in English NIL text and "94(*jiu3 si4* in Chinese Pinyin)" to replace "就是(*jiu4 shi4*, exactly be)" in Chinese NIL text. Such characteristics pose problems to conventional natural language processing (NLP) tools in handling NIL text. For example, the NIL term "细 8 细(*xi4 ba1 xi4*)", which is used to replace "是不是 (be or be not)", is segmented to three common words, i.e. "细(slim)", " 8 (eight)" and "细", with ICTCLAS tool [6]. It can be concluded that without identifying the expression, further Chinese text processing techniques are not able to produce reasonable result. It is suggested that NIL terms like "细 8 细" should be treated as new words. However, we find a large number of NIL terms are included by standard dictionaries, while they are used to express new meanings. For example, "稀饭(*xi1 fan4*)" appears in standard dictionaries with the meaning of Chinese porridge. But in NIL text it represents "喜欢(*xi3 huan1*)" which means "like". The issue that concerns us is that the expressions like "稀饭" may also appear in NIL text with their formal meaning. This is a typical ambiguity problem for natural language processing. Besides errors caused by lexical ambiguity, other errors also occur in processing NIL phrases. In order to obtain a better quality, adjusted or new techniques are required to extract and normalize the NIL expressions before conventional NLP tools are deployed in NIL text processing.

NIL is a domain-specific human language that changes rapidly with new NIL terms and NIL phrasal patterns created every day. Preliminary experiments in [7] reveal that knowledge based approach, i.e. pattern matching, exhibits poor adaptivity when processing unseen NIL expressions. Instead, corpus based machine learning approaches appear to be more robust in processing text. This results in our intention to construct an annotated NIL corpus which is used specially for developing and evaluating techniques in extraction and normalization of NIL expressions. In our research we target at two types of NIL expressions, i.e. NIL terms that hold anomalous morphological forms or meanings compared to formal human language in Chinese NIL text.

Obtaining large scale real chat text is difficult due to privacy concerns. Fortunately, we have located BBS chat texts within “大嘴区(da4 zui3 qu1, free chat zone)” in YESKY BBS system (http://bbs.yesky.com/bbs/) which exhibits remarkable chat characteristics and contains a vast amount of NIL expressions. We download BBS chat texts posted from December 2004 to July 2005 in this zone. We finally collected 12,112 pieces of NIL text containing 92,314 words and 12,983 NIL expressions.

We seek to develop a two-stage annotation approach in this paper to minimize human involvement in creating an annotated NIL corpus. In the first stage, the first 1000 pieces of raw NIL text pieces regarding timestamp are annotated under the specification of NILEML on the annotation platform. In the second stage, an automated annotation module is incrementally trained on annotated NIL text pieces and applied to identify and annotate NIL expressions in the next 1000 NIL text pieces. We argue that the efficiency of corpus annotation can be greatly improved because most annotated NIL expressions can be annotated automatically and some unannotated NIL expressions can be recognized by this module. Therefore, human annotators merely concentrate on 1) justifying the automated annotation, and 2) identifying and annotating the unrecognized NIL expressions. Moreover, the annotation platform can be easily adapted to changed/new domain because the automated annotation module can be incrementally trained with the annotated NIL text pieces in the changed/new domain.

The remaining sections of this paper are organized as follows. The NILEML annotation scheme is first presented in Section 2. In Section 3 we describe the annotation components in this corpus annotation task. In Section 4 we describe the two-stage incremental annotation approach. Several experiments are presented and discussed in Section 5 as well as discussions. We address related works in Section 6 and conclude this paper in Section 7.

2 NILEML corpus annotation scheme

In this paper a NIL corpus annotation scheme, i.e. NILEML, is devised based on XML. In NILEML scheme, we define NILEML tag to describe the NIL text documents and NILEX tag to describe NIL expressions. NILEX entails attributes of NIL expression such as text string, class, equivalent normal language text, part of speech (POS) tag, segments if it is a phrase, POS tags for all segments, and Chinese Pinyin. We present descriptions for the attributes in Figure 1.

For checking the XML tagging syntax conformity, a document type definition (DTD) file is created to properly specify NILEML and NILEX tag set.

Two issues should be carefully considered in defining tag set for NILEML. The first issue is completeness of the tag set. NILEML is currently a task-orientated annotation scheme. Thus only attributes used in recognition task are configured in the annotation scheme. At the same time, we try to cover most commonly used linguistic attributes such as word segments and POS tag. Justification of value set for each attribute is the second issue we should address. Due to observation limit, we are only able to define values we’ve encountered. For example, we define value set for class attribute to be {'A', 'F', 'H', 'T', 'O'} based on observation of 13,068 NIL expressions. There is no doubt that new values will appear in the future annotation. The treatment is that we either cover them by 'O' or represent them by a new value. XML allows that an annotation scheme can be extended easily.

```
attributes ::= nid string class normal
pos [segments] [posseg] [pinyin]

nid ::= n<integer>
string ::= CDATA
  ; string of the NIL expression
class ::= CDATA
{class ::= 'A'|'F'|'H'|'T'|'O'}
  ; class of the NIL expression
  ; A = Abbreviation
  ; F = Foreign expressions
  ; H = Homophony
  ; T = Transliteration
  ; O = Other classes
normal := CDATA  ; normal text for
  ; the NIL expression
pos := CDATA
{pos := 'NOUN'|'PRON'|'VERB'
|'ADJ'|'ADV'|'NUMBER'|'UNIT'
|'PREP'|'CONJ'|'AUX'|'EXCL'}
  ; POS tag for the NIL expression
segments := CDATA  ; segments for
  ; the NIL expression
posseg := CDATA
{posseg := 'pos|+'}
  ; POS tag list for the segments
  ; for the NIL expression
pinyin := CDATA ; Chinese pinyin
  ; for the NIL expression
```

Figure 1. Definition of NILEX attributes.

3 Annotation components

3.1 Computer aided annotation platform

Defining attributes of NIL expressions must be conducted by experts who are familiar with NIL language even though a XML-tagged text can be created using text editors. However, not all experts are familiar with XML tags. To help the human annotators, we develop a GUI-based computer aided annotation platform.

On the platform, human annotators can be concentrating on defining linguistic attributes for the NIL expressions while the NILEML tag set is automatically managed beneath the interface. For example, when the annotator select HOMOPHONY for the NIL expression in the dropdown list for class attribute, the class attribute, namely “class=‘H’”, is inserted in to the current NILX tag automatically.

On the platform ICTCLAS tool is integrated to provide conventional NLP functionalities such as word segmentation and POS tagging. To produce Chinese Pinyin automatically, a Chinese Pinyin transcription tool is developed based on CEDICT Chinese-English dictionary [8].

3.2 Automated annotation module

The automated annotation module integrates a SVM classifier which is trained on the annotated NIL expressions and used to identify NIL expressions in new chat text automatically. When a NIL term is recognized, a search action is executed to check whether it appears in the annotated NIL corpus. The whole NILEX tag is duplicated as the tag of the identified NIL term if it already exists. Otherwise, an empty NILEX tag will be created by the platform and a human annotator is required to specify its attributes.

NIL term recognition is seen as a binary classification task in the annotation module. We use the SVM^{light} [9] in our SVM implementation. Features we consider for each NIL expression in SVM classification are listed as follow.

1) The occurrence of each NIL term when

- its string appears in any word bi-grams or tri-grams,
- its POS tag appears in any POS tag bi-grams or tri-grams;
- its POS tags for segments appear in any POS tag bi-grams or tri-grams;

2) The Boolean value that indicates whether a NIL term

- is a number (Chinese or Arabic);
- contains merely Latin capitals;
- contains more than two standard Chinese words;
- contains punctuations;
- mixes Chinese character and number;
- mixes Chinese character and Latin characters;

An input NIL text is first segmented using ICTCLAS tool. We then use the SVM classifier to process all sequential combinations of the segments. For example, the NIL text “这个人８错(zhe4 ge4 ren2 ba1 cuo4, This people is not bad)” is first segmented to “这个|人|８|错”. Ten sequential combinations are processed by the SVM classifiers, i.e. {“这个”, “这个人”, “这个人８”, “这个人８错”, “人”, “人８”, “人８错”, “８”, “８错”, “错”}. For this case, “８错” is identified as a NIL term by the SVM classifier. To reduce computational complexity, we choose to combine up to four standard words in NIL term recognition.

Note that some recognized NIL expressions by the SVM classifier are identical to those annotated in the NIL corpus. The SVM classifier is also able to recognize NIL expressions with similar compositions to those annotated in the NIL corpus. This is largely due to the features that describe compositional characteristics.

4 Two-stage incremental annotation approach

We propose a two-stage incremental annotation approach in this paper. In the first stage, we sort the raw NIL text pieces according to their timestamp. We then split the sorted NIL text pieces into several blocks in which each block contains 1000 NIL text pieces. As the last block might contain less than 1000 NIL text pieces, we merge the last two blocks into one. The 1000 NIL text pieces in the first block are annotated by the human annotators on the annotation platform.

In the second stage an automated annotation module is developed. We first train the module on the annotated NIL text to obtain capability of NIL term recognition. With the module, NIL text pieces in the rest blocks are annotated incrementally. The workflow for the incremental annotation is shown in Figure 2.

4.1 Stage I: manual annotation

In this stage the first block of 1000 pieces of raw NIL text are annotated under the specification of NILEML by human annotators on the annotation platform.

NIL expressions are identified from NIL text manually. Attributes for NIL expressions are specified by human annotators. To improve efficiency and quality, several conventional NLP tools are integrated to produce some attributes automatically.

For example, the annotators are required to assign one of four classes (i.e., 'A', 'F', 'H' and 'T') to each NIL expression. The equivalent formal language text for each NIL expression is also defined by them manually. Word segments and POS tags can be produced by ICTCLAS tool automatically. A Chinese Pinyin transcription tool is employed to produce standard Chinese Pinyin for Chinese characters.

Coordination between annotators in this stage is plentiful because every NIL expression is annotated for the first time. The annotators have to negotiate with each other on whether a piece of text string is a NIL expression and how its attributes are specified.

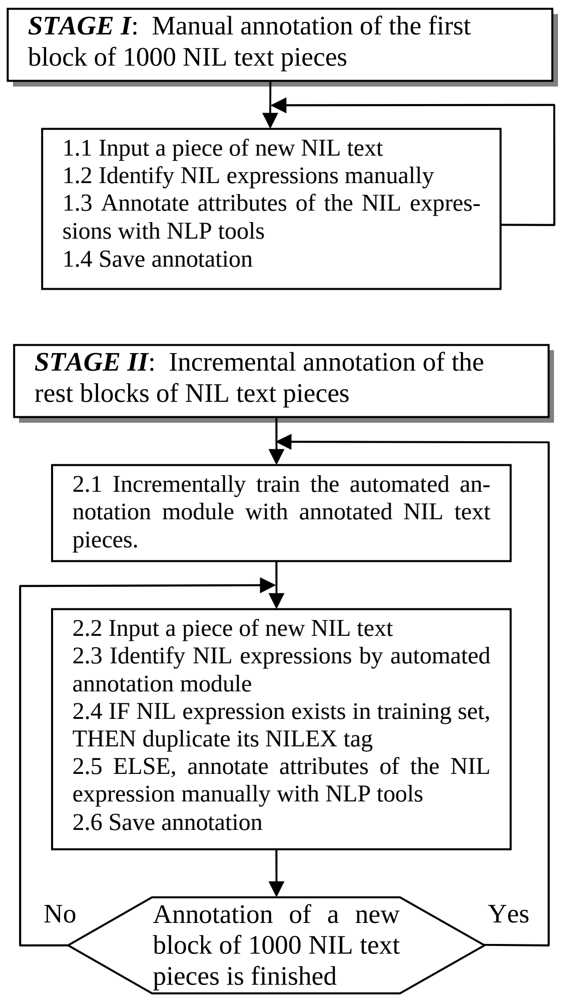
STAGE I: Manual annotation of the first block of 1000 NIL text pieces
1.1 Input a piece of new NIL text
1.2 Identify NIL expressions manually
1.3 Annotate attributes of the NIL expressions with NLP tools
1.4 Save annotation
STAGE II: Incremental annotation of the rest blocks of NIL text pieces
2.1 Incrementally train the automated annotation module with annotated NIL text pieces.
2.2 Input a piece of new NIL text
2.3 Identify NIL expressions by automated annotation module
2.4 IF NIL expression exists in training set, THEN duplicate its NILEX tag
2.5 ELSE, annotate attributes of the NIL expression manually with NLP tools
2.6 Save annotation
No
Annotation of a new block of 1000 NIL text pieces is finished
Yes

Figure 2. Workflow of the two-stage NIL corpus annotation

4.2 Stage II: incremental annotation

In this stage we develop an automated annotation module based on SVM pattern recognition technique. We train the module on the annotated NIL text pieces and use it to annotate NIL text pieces in every next block. In other words, at the very beginning of the second stage, the module is trained on the 1000 annotated NIL text pieces and applied to annotate the second block of 1000 NIL text pieces. When the second 1000 NIL text pieces are successfully annotated, we obtain 2000 annotated NIL text pieces in total. We train the module incrementally on the 2000 annotated NIL text pieces and apply the updated module to the third block of 1000 NIL text pieces. The incremental annotation process is repeated until all blocks of NIL text pieces are annotated.

It is not likely that all NIL expressions in the incoming block of unannotated NIL text pieces can be identified correctly because many unannotated NIL expressions are often found due to the limited scope of the training data. We devise the annotation method as follows. The module first recognizes NIL expressions in the to-be-annotated NIL text pieces. If an identified NIL expression appears within the annotated NIL corpus, the previous NILEX tag is duplicated by the module as the tag for the recognized NIL expression. Otherwise, an empty NILEX tag will be created and the human annotator is prompted on the platform to specify its attributes. Thus human annotators' part in the annotation work is summarized to be, 1) justifying the automated annotation, 2) annotating the identified unannotated NIL expressions, and 3) identifying and annotating the unrecognized unannotated NIL expressions manually.

We believe that efficiency of corpus annotation can be significantly improved because a large number of annotated NIL expressions can be duplicated automatically by the annotation module. Notably, the annotation platform is made adaptive in identifying unannotated NIL expressions because the automated annotation module can be incrementally trained with annotated NIL text pieces.

4.3 Annotation consistency

Consistency is a large problem for each annotation task. It entails inter-annotator agreement (i.e., one sentence is annotated by two annotators equally) and intra-annotator consistency (i.e., annotations for same sentences are equally by one annotator). Consistency is usually maintained during the whole annotation process in which several annotators are possibly involved. Usefulness of a corpus relies highly on consistency in training or testing automatic methods. To guarantee a satisfactory annotation consistency, we define some guiding annotation principles as follow.

- The annotators are strictly required to negotiate with each other to produce an agreed annotation for each new NIL expression.
- The annotators are strictly required to duplicate previous annotation for the NIL expression in existence.
- When suggested, revision should be agreed by all annotators.
- When revision is finally conducted, annotation for same NIL expressions must be revised at the same time.

The restrictions are setup to avoid inconsistency between annotators during corpus annotation. High intra-consistency and inter-annotator agreement are thus obtained to assure the quality of the annotation in terms of consistency.

5 Experiments

To evaluate how significantly the automatic annotation module improves the efficiency of NIL corpus annotation, several experiments are conducted to simulate the aforementioned incremental annotation process and reproduce the performance of the module over different versions of the training set and the test set.

5.1 Experiment setup

Similar to the incremental annotation process, we use the time-based incremental training/test data split strategy according to the timestamp of the NIL text pieces. In the annotation-ready NIL corpus we currently have 12,112 annotated NIL text pieces sorted with timestamp from December 2004 to June 2005. We start the experiments from training the module on the first block of 1000 annotated NIL text pieces and testing it with the second block of 1000 NIL text pieces. Then we train the module on the first two blocks and test it on the third block. We repeat such training and test processes until in round 11 we use the first 11 blocks as the training set and the rest 1112 unannotated NIL text pieces as the test set. Training/test data for all experiments are presented in Table 1.

Table 1. Training/test data description.

Round. No.	# of training NIL Exp.	# of test NIL Exp.	# of annotated test NIL Exp.	# of unannotated test NIL Exp.
1	996	997	414	583
2	1992	998	494	504
3	2989	1000	564	436
4	3988	997	583	414
5	4984	1001	648	353
6	5984	998	702	296
7	6981	995	713	282
8	7975	992	764	228
9	8966	998	791	207
10	9963	996	861	135
11	11956	1112	999	113

Coverage curves for annotated and unannotated NIL expressions are presented Figure 3. We find percentage of unannotated NIL expressions decreases from 58.5% in round 1 to 10.2% in round 11.

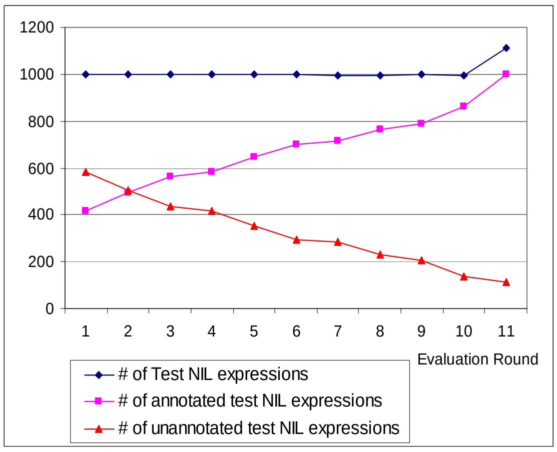

Figure 3. Coverage curves for annotated and unannotated NIIL expressions in the 11 test sets.

We use precision, recall and F-1 measure to present quality of NIL expression recognition. The precision is defined as the percentage of NIL expressions recognized correctly in all recognized NIL expressions. The recall is defined to be the percentage of NIL expressions recognized correctly in those annotated by human annotators.

5.2 Results

We run the evaluation process on the eleven versions of training/test set. The overall experimental results are presented in Table 2.

Table 2. Overall experimental results.

Round No.	Precision	Recall	F-1
1	68.6	58.3	63.0
2	72.1	63.8	67.7
3	75.1	64.7	69.5
4	77.2	66.3	71.4
5	78.6	71.9	75.1
6	80.5	74.4	77.3
7	82.0	78.5	80.2
8	84.2	80.1	82.1
9	85.8	82.5	84.1
10	87.7	83.9	85.8
11	88.7	86.5	87.6

It's natural that we split the test NIL expressions into the annotated and the unannotated. We refer the annotated NIL expressions to identical NIL expressions in the training NIL corpus, and the unannotated NIL expressions to those not identical but with similar compositions to the annotated NIL expressions in the training NIL corpus. We present results for the annotated and unannotated NIL expressions in Table 3 and Table 4 respectively.

Table 3. Overall experimental results for annotated NIL expressions.

Round No.	Precision	Recall	F-1
1	88.9	77.5	82.8
2	87.9	82.4	85.0
3	89.4	78.1	83.4
4	89.7	81.0	85.1
5	88.4	87.3	87.9
6	88.0	87.9	88.0
7	89.9	92.5	91.2
8	89.1	91.8	90.4
9	90.4	90.7	90.6
10	90.8	89.1	89.9
11	91.0	89.6	90.3

Table 4. Overall experimental results for unannotated NIL expressions.

Round No.	Precision	Recall	F-1
1	54.2	45.2	49.3
2	56.8	47.6	51.8
3	56.7	48.0	51.9
4	59.7	48.0	53.2
5	60.6	48.7	54.0
6	62.5	49.2	55.0
7	62.1	50.6	55.7
8	67.5	51.2	58.2
9	68.1	56.4	61.7
10	67.4	56.2	61.3
11	69.2	62.9	65.9

5.3 Discussion I: recognition quality

We present the quality curves for precision, recall and F-1 measure in identifying all NIL expressions in Figure 4. It is observed that when the volume of more training data increases, the overall quality is improved gradually. Note that evaluation was carried out within the same domain. This undoubtedly leads to high quality in the last several experiment rounds.

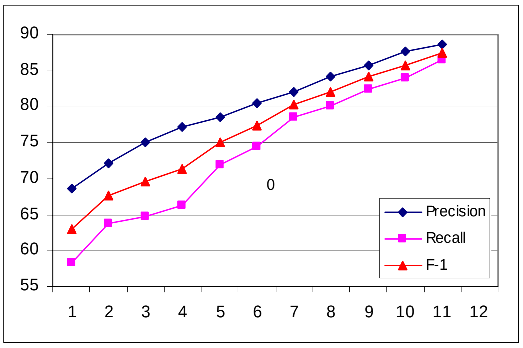

Figure 4. Quality curves for overall recognition.

We present quality curves in identifying annotated and unannotated NIL expressions in Figure 5 and Figure 6 respectively.

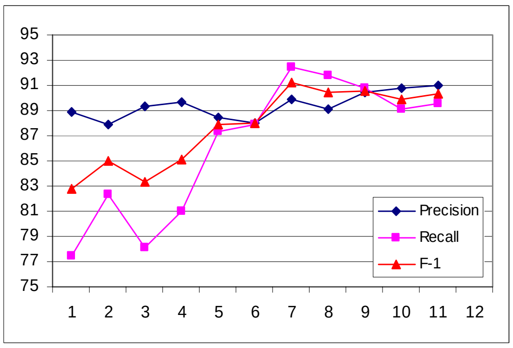

Figure 5. Quality curves for recognition of annotated NIIL expressions.

Curves in Figure 5 show that precision in identifying annotated NIL expressions is relatively stable at around 90% in all experiments. This reveals that most annotated NIL expressions can be correctly recognized. However, it is another story for recall. In the first five rounds it climbs up from around 77% to 87%. This is reasonable because more training data normally leads to higher recall. Since in the last six rounds recall remains relatively stable, we may conclude that training data in round 5 is probably sufficient in identifying annotated NIL expressions.

Identifying unannotated NIL expressions is much more difficult than identifying annotated ones with the SVM classifier. However, very encouragingly, the quality catches up when more training data is available according to Figure 6. Coverage curves for unannotated NIL expressions in Figure 3 show that training data in round 11 is near to a sufficient volume in identifying unannotated NIL expressions.

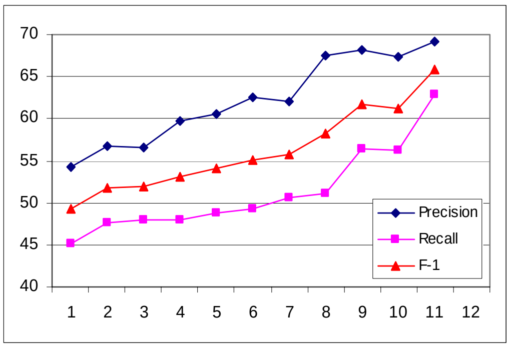

Figure 6. Quality curves for recognition of unannotated NIIL expressions.

5.4 Discussion II: annotation efficiency

In our NIL corpus annotation experience we find that the annotation efficiency can be improved in two manners. On the one hand, around 90% annotated NIL expressions can be identified from NIL text correctly. Their NILEX tags can be duplicated without any changes. Human annotators' efforts can be reduced to confirming the automated annotation and identifying the unrecognized NIL expressions. A large volume of repetitive annotation work is therefore avoided.

On the other hand, the SVM classifier produces increasingly better quality in terms of correctly recognizing unannotated NIL expressions. For example, around 70% unannotated NIL expressions are identified correctly. The recognition output is helpful to alarm the human annotators, thus alleviate their work in picking out potential NIL expressions quickly. Efforts thus can be saved in carrying out this painstaking work.

Time used in annotating NIL text pieces in every block is presented in Table 5.

Table 5. Annotation time (minutes) used in 12 annotation rounds.

Round. No.	Annotated NIL Exp.	Unannotated NIL Exp.	Total minutes
0	0	5031	5031
1	184	1580	1764
2	221	1431	1652
3	264	1236	1500
4	253	1236	1489
5	272	1070	1342
6	319	925	1244
7	332	876	1208
8	326	770	1096
9	348	705	1053
10	390	455	845
11	449	391	840

Time used in manually annotating the first block of 1000 NIL text pieces in round 0 is 5031 minutes, namely around 5 minutes per NIL expression. According to Table 5, annotation time is reduced to 0.76 minute per NIL expression. It is thus proved that the efficiency is improved by 85.0% in annotating the last 1112 NIL text pieces.

5.5 Error analysis

We summarize two types of typical recognition errors occurring in our experiments.

Err.1 Ambiguous NIL Expressions

Chat text always contains common words with same characters as some NIL expressions. For example, when used in “答谢粉丝(da2 xie4 fen3 si1, thank the fans)”, “粉丝(fen3 si1, vermicelli made from bean starch)” equals to ‘fans’. But when used in “今天吃粉丝(jin1 tian1 chi1 fen3 si1, eat vermicelli today)”, it is just a kind of food material. Such ambiguity also occurs frequently for the numbers. Forty errors with this type happened in our experiment round 11.

Err.2 Unannotated NIL Expressions

New NIL expressions come into birth very quickly. When the unannotated NIL expressions provide no clue (e.g., satisfying any feature), the SVM classifier is incapable in those cases. Five errors with this type happened in our experiment round 11 including “盒饭(he2 fan4, takeaway food)” which represents ‘fans of He Jie’ (He Jie is a Chinese girl who got widely known recently in a TV show) .

6 Related works

Corpus annotation is a prerequisite for many machine learning methods in NLP but suffers from high cost and inter-annotator inconsistency. An interactive annotation approach is devised in [10] in tagging and parsing the NEGRA corpus. Suggestions are produced automatically by a Cascaded Markov Models. The model is able to calculate reliability of the suggestions, and the annotator is prompted for confirmation or correction of the unreliable assignments. Such a semi-automatic process facilitates a very rapid and efficient annotation. However, tagging and parsing capability remains static during the corpus annotation process. We argue that the annotated text can be very helpful in improving tagging and parsing capability.

The feasibility of incremental linguistic annotation is examined in [11]. The article encourages reuse of annotated corpora already in existence and urge sufficient care should be taken with ambiguity, consistency and correctness in the incremental annotation. It is also argued that the feasibility of such an increase depends heavily on the way in which the incremental annotation is implemented. The two-stage incremental annotation approach is enlightened by the principle of incremental annotation. However, we discard the ambitious solution, i.e. fully automated incremental annotation. In our approach, we reduce the amount of human involvement as much as possible. Human efforts are expected to be concentrating on justifying ambiguity, consistency and correctness.

7 Conclusions

We propose a two-stage annotation approach in this paper. The first block of 1000 NIL text pieces regarding timestamp are annotated by human annotators in the first stage. In the second stage an automated annotation module is incrementally trained on all annotated NIL text available and applied to identify and to annotate NIL expressions in every next block of 1000 NIL text pieces. Two conclusions can be drawn from our experiments. One, with increasing volume of annotated NIL text pieces, quality of automated annotation of incoming NIL text pieces can be improved gradually to around 88.7% in terms of precision. Two, the efficiency of corpus annotation is improved by 85.0% with the automated annotation module because more than 90% annotated NIL expressions and more than 50% unannotated NIL expressions can be annotated correctly with the annotation module.

References

[1] German News: Germans are world SMS champions, 8 April 2004, http://www.expatica.com/source/site_article.asp?subchannel_id=52&story_id=6469.

[2] Melanie Heard-White, Gunter Saunders, Anita Pincas. 2004. Report into the use of CHAT in education. Final report for project of Effective use of CHAT in Online Learning, Institute of Education, University of London.

[3] Gianforte, G.. 2003. From Call Center to Contact Center: How to Successfully Blend Phone, Email, Web and Chat to Deliver Great Service and Slash Costs. RightNow Technologies.

[4] McCullagh, D.. 2004. Security officials to spy on chat rooms. News provided by CNET Networks. November 24, 2004.

[5] Finkelhor, D., K. J. Mitchell, and J. Wolak. Online Victimization: A Report on the Nation's Youth. Alexandria, Virginia: National Center for Missing & Exploited Children, 2000, page ix.

[6] Zhang, Z., H. Yu, D. Xiong and Q. Liu. 2003. HMM-based Chinese Lexical Analyzer ICTCLAS. SIGHAN'03 within ACL'03, pp. 184-187.

[7] Xia, Y. and K.-F. Wong. 2005. NIL is not Nothing: Recognition of Chinese Network Informal Language Expressions, 4th SIGHAN Workshop on Chinese Language Processing at IJCNLP'05.

[8] Denisowski, P.. CEDICT: Chinese - English Dictionary. http://www.mandarintools.com/ cedict.html.

[9] Joachims, T. 1998. Text categorization with Support Vector Machines: Learning with many relevant features. ECML'98, pp. 137-142.]

[10] Thorsten, B. and P. Oliver. Interactive Corpus Annotation. In Proc. of LREC 2000. 31 May – 2 June, 2000. Athens, Greece.

[11] Hans van Halteren. The Feasibility of Incremental Linguistic Annotation. http://www.cs.queensu.ca/ achallc97/papers/p019.html.