Improving Context Vector Models by Feature Clustering for Automatic Thesaurus Construction

Jia-Ming You
Institute of Information Science
Academia Sinica
swimming@hp.iis.sinica.edu.tw

Keh-Jiann Chen
Institute of Information Science
Academia Sinica
kchen@iis.sinica.edu.tw

Abstract

Thesauruses are useful resources for NLP; however, manual construction of thesaurus is time consuming and suffers low coverage. Automatic thesaurus construction is developed to solve the problem. Conventional way to automatically construct thesaurus is by finding similar words based on context vector models and then organizing similar words into thesaurus structure. But the context vector methods suffer from the problems of vast feature dimensions and data sparseness. Latent Semantic Index (LSI) was commonly used to overcome the problems. In this paper, we propose a feature clustering method to overcome the same problems. The experimental results show that it performs better than the LSI models and do enhance contextual information for infrequent words.

1 Introduction

Thesaurus is one of the most useful linguistic resources. It provides information more than just synonyms. For example, in WordNet (Fellbaum, 1998), it also builds up relations between synonym sets, such as hyponym, hypernym. There are two Chinese thesauruses Cilin(1983) and Hownet[1]. Cilin provides synonym sets with simple hierarchical structure. Hownet uses some primitive senses to describe word meanings. The common primitive senses provide additional relations between words implicitly. However, many words occurred in contemporary news corpora are not covered by Chinese thesauruses.

Therefore, we intend to create a thesaurus based on contemporary news corpora. The common steps to automatically construct a thesaurus include a) contextual information extraction, b) finding synonym words and c) organizing synonym words into a thesaurus. The approach is based upon the fact that word meaning lays on its contextual behavior. If words act similarly in context, they may share the same meaning. However, the method can only handle frequent words rather than infrequent ones. In fact most of vocabularies occur infrequently, one has to discover extend information to overcome the data sparseness problem. We will introduce the conventional approaches for automatic thesaurus construction in section 2. Follow a discussion about the problems and solutions of context vector models in section 3. In section 4, we use two performance evaluation metrics, i.e. discrimination and nonlinear interpolated precision, to evaluate our proposed method.

2 Conventional approaches for automatic thesaurus construction

The conventional approaches for automatic thesaurus construction include three steps: (1) Acquire contextual behaviors of words from corpora. (2) Calculate the similarity between words. (3) Finding similar words and then organizing into a thesaurus structure.

2.1 Acquire word sense knowledge

One can model word meanings by their co-occurrence context. The common ways to extract co-occurrence contextual words include simple window based and syntactic dependent based (You, 2004). Obviously, syntactic dependent relations carry more accurate information than window based. Also, it can bring additional information, such as POS (part of speech) and semantic roles etc. To extract the syntactic de-

[1] *http://www.HowNet.com(Dong Zhendong, Dong Qiang:HowNet)*

Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing, pages 1–8,
Sydney, July 2006.

pended relation, a raw text has to be segmented, POS tagged, and parsed. Then the relation extractor identifies the head-modifier relations and/or head-argument relations. Each relation could be defined as a triple (w, r, c), where w is the thesaurus term, c is the co-occurred context word and r is the relation between w and c.

Then context vector of a word is represented differently by different models, such as: tf, weight-tf, Latent Semantic Indexing (LSI) (Deerwester, S.,et al., 1990) and Probabilistic LSI (Hofmann, 1999). The context vectors of word *x* can be express by:

a) tf model: word $x = \{tf_1^x, tf_2^x, ..., tf_n^x\}$,where tf_i^x is the term frequency of the *ith* context word when given word x.

b) weight-tf model: assume there are *n* contextual words and *m* target words. word *x*=

$$\{tf_1^x \times weight1, tf_2^x \times weight2 ... tf_n^x \times weight n\}$$

,where $weight_i$, we used here, is defined as [log*m*-entropy(word*i*)]/log*m*

$$entropy(wordi) = \sum_{k=1}^{m} - p(word_k^i) \log p(word_k^i); \quad p(word_k^i)$$

is the co-occurrence probability of word*k* when given word*i*.

c) LSI or PLSI models: using tf or weighted-tf co-occurrence matrix and by adopting LSI or PLSI to reduce the dimension of the matrix.

2.2 Similarity between words

The common similarity functions include
a) Adopting simple frequency feature, such as cosine, which computes the angle between two context vectors;

$$\cos(x, y) = \frac{x \bullet y}{\|x\| \times \|y\|}$$

b) Represent words by the probabilistic distribution among contexts, such as Kull-Leiber divergence (Cover and Thomas, 1991).

The first step is to convert the co-occurrence matrix into a probabilistic matrix by simple formula.

$$wordx = p = \{p_1^x, p_2^x, ...p_n^x\}, p_i^x = tf_i^x \Big/ \sum_{k=1}^{n} tf_k^x$$

$$wordy = q = \{q_1^x, q_2^x, ...q_n^x\}, q_i^x = tf_i^y \Big/ \sum_{k=1}^{n} tf_k^y$$

Then calculate the distance between probabilistic vectors by sums up the all probabilistic difference among each context word so called cross entropy.

$$KL\ Distance: KL(p, q) = \sum_{i=1}^{n} p(i) \bullet \log_2 p_i \Big/ q_i$$

Due to the original KL distance is asymmetric and is not defined when zero frequency occurs. Some enhanced KL models were developed to prevent these problems such as Jensen-Shannon (Jianhua, 1991), which introducing a probabilistic variable m, or α -Skew Divergence (Lee, 1999), by adopting adjustable variable α. Research shows that Skew Divergence achieves better performance than other measures. (Lee, 2001)

$$D(SkewDivergence) = S_x^y a = KL(x \parallel ax + (1 - a)y)$$

$$D(Jensen - Shannon) = JS(x, y) = \{KL(x \parallel m) + KL(y \parallel m)\} / 2,$$
$$m = (x + y) / 2$$

To convert distance to similarity value, we adopt the formula inspired by Mochihashi, and Matsumoto 2002.

$$similarity(wordx, wordy) = \exp\{-l \cdot distance(x, y)\}$$

2.3 Organize similar words into thesaurus

There are several clustering methods can be used to cluster similar words. For example, by selecting N target words as the entries of a thesaurus, then extract top-n similar words for each entry; adopting HAC(Hierarchical agglomerative clustering, E.M. Voorhees,1986) method to cluster the most similar word pairs in each clustering loop. Eventually, these similar words will be formed into synonyms sets.

3 Difficulties and Solutions

There are two difficulties of using context vector models. One is the enormous dimensions of con-

textual words, and the other is data sparseness problem. Conventionally LSI or PLSI methods are used to reduce feature dimensions by mapping literal words into latent semantic classes. The researches show that it's a promising method (April Kontostathis, 2003). However the latent semantic classes also smooth the information content of feature vectors. Here we proposed a different approach to cope with the feature reduction and data sparseness problems.

3.1 Feature Clustering

Reduced feature dimensions and data sparseness cause the problem of inaccurate contextual information. In general, one has to reduce the feature dimensions for computational feasibility and also to extend the contextual word information to overcome the problem of insufficient context information.

In our experiments, we took the clustered-feature approaches instead of LSI to cope with these two problems and showed better performances. The idea of clustered-feature approaches is by adopting the classes of clustering result of the frequent words as the new set of features which has less feature dimensions and context words are naturally extend to their class members. We followed the steps described in section 2 to develop the synonyms sets. First, the syntactic dependent relations were extracted to create the context vectors for each word. We adopted the skew divergence as the similarity function, which is reported to be the suitable similarity function (Masato, 2005), to measure the distance between words.

We used HAC algorithm to develop the synonyms classes, which is a greedy method, simply to cluster the most similar word pairs at each clustering iteration.

The HAC clustering process:

While the similarity of the most similar word pair (wordx, wordy) is greater than a threshold ε

then cluster wordx, wordy together and replace it with the centroid between wordx and wordy

Recalculate the similarity between other words and the centroid

3.2 Clustered-Feature Vectors

We obtain the synonyms sets S from above HAC method. Let the extracted synonyms sets $S = \{ S^1, S^2, \ldots S^R\}$ which contains R synonym classes; S^i_j stands for the *jth* element of the *ith* synonym class; the *i*th synonym class S^i contains Q_i elements.

$$S = \begin{bmatrix} S_1^1 & S_2^1 & \ldots & S_{Q1}^1 \\ S_1^2 & S_2^2 & \ldots & S_{Q2}^2 \\ \ldots & \ldots & \ldots & \ldots \\ S_1^R & S_2^R & \ldots & S_{QR}^R \end{bmatrix}$$

The feature extension processing transforms the coordination from literal words to synonyms sets. Assume there are *N* contextual words $\{C_1, C_2, \ldots C_N\}$, and the first step is to transform the context vector of of C_i to the distribution vector among S. Then the new feature vector is the summation of the distribution vectors among S of its all contextual words.

The new feature vector of $word_j$ =

$$\sum_{i=1}^{N} tf_i^j \times \text{Distribution\_Vector\_among\_S}(C_i)$$

,where tf_i^j is the term frequency of the context word C_i occurs with $word_j$.

$\text{Distribution\_Vector\_among\_S}(C_i) = \left\{ P_i^{S_1}, P_i^{S_2}, .. P_i^{S_R} \right\}$,

, where $P_i^{S_j} = \dfrac{\sum_{q=1}^{Qj} freq(S_q^j, Ci)}{freq(\text{Ci})}$, means the distribution of context words of *Ci* at the *jth* synonyms S^j.

Due to the transformed coordination no longer stands for either frequency or probability, we use simple cosine function to measure the similarity between these transformed clustered-feature vectors.

4 Evaluation

To evaluate the performance of the feature clustering method, we had prepared two sets of testing data with high and low frequency words respectively. We want to see the effects of feature reduction and feature extension for both frequent and infrequent words.

4.1 Discrimination Rates

The discrimination rate is used to examine the capability of distinguishing the correlation between words. Given a word pair (*wordi,wordj*), one has to decide whether the word pair is similar or not. Therefore, we will arrange two different word pair sets, related and unrelated, to estimate the discrimination. By given the formula below

$$\text{Discrimination rate} = \frac{1}{2}\left(\frac{na}{Na} + \frac{nb}{Nb}\right)$$

,where Na and Nb are respectively the numbers of synonym word pairs and unrelated word pairs. As well as, na and nb are the numbers of correct labeled pairs in synonyms and unrelated words.

4.2 Nonlinear interpolated precision

The Nap evaluation is used to measure the performance of restoring words to taxonomy, a similar task of restoring words in WordNet (Dominic Widdows, 2003).

The way we adopted Nap evaluation is to reconstruct a partial Chinese synonym set, and measure the structure resemblance between original synonyms and the reconstructed one. By doing so, one has to prepare certain number of synonyms sets from Chinese taxonomy, and try to reclassify these words.

Assume there are *n* testing words distributed in *R* synonyms sets. Let R_1^i stands for the represented word of the *ith* synonyms set. Then we will compute the similarity ranking between each represented word and the rest *n-1* testing words. By given formula

$$\text{NAP} = \frac{1}{R}\sum_{i=1}^{R}\sum_{j=1}^{n-1}\frac{Z_j^i}{j}\left(1+\sum_{k=1}^{j-1}Z_k^i\right)$$

S_j^i represents the *jth* similar word of R_1^i among the rest *n-1* words

$$Z_j^i = \begin{cases} 1, \text{if } S_j^i \text{ and } R_1^i \text{ are synonym} \\ 0 \end{cases}$$

The NAP value means how many percent synonyms can be identified. The maximum value of NAP is 1, means the extracted similar words are exactly match to the synonyms.

5 Experiments

The context vectors were derived from a 10 year news corpus from The Central News Agency. It contains nearly 33 million sentences, 234 million word tokens, and we extracted 186 million syntactic relations from this corpus. Due to the low reliability of infrequent data, only the relation triples (w, r, c), which occurs more than 3 times and POS of w and c must be noun or verb, are used. It results that nearly 30,000 high frequent nouns and verbs are used as the contextual features. And with feature clustering[2], the contextual dimensions were reduced from 30,988 literal words to 12,032 semantic classes.

In selecting testing data, we consider the words that occur more than 200 times as high frequent words and the frequencies range from 40 to 200 as low frequent words.

Discrimination

For the discrimination experiments, we randomly extract high frequent word pairs which include 500 synonym pairs and 500 unrelated word pairs from Cilin (Mei et. al, 1983). At the mean time, we also prepare equivalent low frequency data.

We use a mathematical technique Singular Value Decomposition (SVD) to derive principal components and to implement LSI models with respect to different feature dimensions from 100 to 1000. We compare the performances of different models. The results are shown in the following figures.

discrimination	related recall	unrelated words	discrimination rate
TF	81.20%	84.00%	82.60%
Weight_TF	77%	88.40%	82.70%
Feature Clustering	81.80%	82%	81.90%
SVD 100	60.80%	89.80%	75.30%
SVD200	65.60%	85.80%	75.70%
SVD300	64.20%	90.20%	77.20%
SVD400	69.40%	86.20%	77.80%
SVD500	74%	84.60%	79.30%
SVD600	74.80%	85.40%	80.10%
SVD700	75.20%	84.80%	80%
SVD800	72.40%	89.40%	80.90%
SVD900	70.80%	91%	80.90%
SVD1000	78.60%	83.60%	81.10%

Figure1. Discrimination for high frequent words

The result shows that for the high frequent data, although the feature clustering method did not achieve the best performance, it performances better at related data and a balanced performance at unrelated data. The tradeoffs be-

[2] *Some feature clustering results are listed in the Appendix*

tween related recalls and unrelated recalls are clearly shown. Another observation is that no matter of using LSI or literal word features (tf or weight_tf), the performances are comparable. Therefore, we could simply use any method to handle the high frequent words.

discrimination	related recall	unrelated recall	discrimination rate
TF	53.13%	97.19%	75.16%
Weight_TF	53.13%	97.62%	75.38%
Feature Clustering	**59.18%**	**94.17%**	**76.67%**
SVD 100	64.58%	78.19%	71.38%
SVD200	53.56%	94.17%	73.87%
SVD300	54.43%	94.60%	74.51%
SVD400	53.56%	96.11%	74.84%
SVD500	60.04%	88.98%	74.51%
SVD600	51.40%	95.68%	73.54%
SVD700	54.21%	95.03%	74.62%
SVD800	50.54%	96.98%	73.76%
SVD900	53.56%	96.54%	75.05%

Figure2 Discrimination for low frequent word

For the infrequent words experiments, neither LSI nor weighted-tf performs well due to insufficient contextual information. But by introducing feature clustering method, one can gain more 6% accuracy for the related data. It shows feature clustering method could help gather more information for the infrequent words.

Nonlinear interpolated precision

For the Nap evaluation, we prepared two testing data from Cilin and Hownet. In the high frequent words experiments, we extract 1311 words within 352 synonyms sets from Cilin and 2981 words within 570 synonyms sets from Hownet.

Nap Performance	Cilin	Hownet
TF	38.34%	28.56%
Weight_TF	40.38%	29.00%
Feature Clustering	37.71%	27.61%
SVD100	20.13%	13.12%
SVD200	22.46%	15.73%
SVD300	23.26%	16.21%
SVD400	26.04%	17.28%
SVD500	26.95%	17.37%
SVD600	27.31%	17.93%
SVD700	28.80%	18.58%
SVD800	29.10%	19.14%
SVD900	29.07%	19.48%
SVD1000	29.15%	19.67%

Figure 3. Nap performance for high frequent words

In high frequent experiments, the results show that the models retaining literal form perform better than dimension reduction methods. It means in the task of measuring similarity of high frequent words using literal contextual feature vectors is more precise than using dimension reduction feature vectors.

In the infrequent words experiments, we can only extract 202 words distributed in 62 synonyms sets from Cilin and 1089 words within 222 synonyms sets. Due to fewer testing words, LSI was not applied in this experiment.

Nap Performance	Cilin	Hownet
TF	22.30%	15.60%
Weighted TF	23.60%	17.23%
Feature Clustering	22.56%	16.43%

Figure 4. Nap performance for low frequent words

It shows with insufficient contextual information, the feature clustering method could not help in recalling synonyms because of dimensional reduction.

6. Error Analysis and Conclusion

Using context vector models to construct thesaurus suffers from the problems of large feature dimensions and data sparseness. We propose a feature clustering method to overcome the problems. The experimental results show that it performs better than the LSI models in distinguishing related/unrelated pairs for the infrequent data, and also achieve relevant scores on other evaluations.

Feature clustering method could raise the ability of discrimination, but not robust enough to improve the performance in extracting synonyms. It also reveals the truth that it's easy to distinguish whether a pair is related or unrelated once the word pair shares the same sense in their senses. However, it's not the case when seeking synonyms. One has to discriminate each sense for each word first and then compute the similarity between these senses to achieve synonyms. Because feature clustering method lacks the ability of senses discrimination of a word, the method can handle the task of distinguishing correlation pairs rather than synonyms identification.

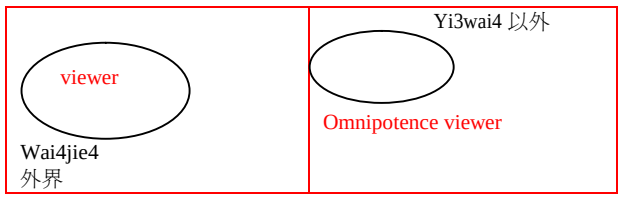
Also, after analyzing discrimination errors made by context vector models, we found that some errors are not due to insufficient contextual information. Certain synonyms have dissimilar contextual contents for different reasons. We observed some phenomenon of these cases:

a) Some senses of synonyms in testing data are not their dominant senses.

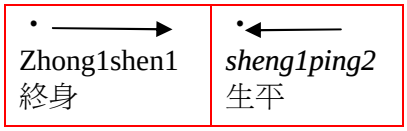
Take *guang1hua2* (光華) for example, it has a sense of "splendid" which is similar to the sense of *guang1mang2* (光芒). *Guang1hua2* and *guang1mang2* are certainly mutually changeable in a certain degree, *guang1hua2jin4shi4 (*光華盡失) and *guang1mang2jin4shi4* (光芒盡失), or *xi2ri4guang1hua2* (昔日光華) and *xi2ri4guang1mang2* (昔日光芒). However, the dominated contextual sense of *guang1hua2* is more likely to be a place name, like *guang1hua2shi4chang3*(光華市場) or *hua1lian2guang1hua2* (花蓮光華) etc[3].

b) Some synonyms are different in usages for pragmatic reasons.

Synonyms with different contextual vectors could be result from different perspective views. For example, we may view *wai4jie4* (外界) as a container image with viewer inside, but on the other hand, *yi3wai4* (以外) is an omnipotence perspective. This similar meaning but different perspective makes distinct grammatical usage and different collocations.



Similarly, *zhong1shen1* (終身) and *sheng1ping2* (生平) both refer to "life-long time". *zhong1shen1* explicates things after a time point, which differs from *sheng1ping2*, showing matters before a time point.



c) **Domain specific usages**.

For example, in medical domain news ,*wa1wa1* (娃娃) occurs frequently with *bo1li2* (玻璃) refer to kind of illness. Then the corpus reinterpret *wa1wa1* (娃娃) as a sick people, due to it occurs with medical term. But the synonym of *wa1wa1* (娃娃), xiao3peng2you3(小朋友) stands for money in some finance news. Therefore, the meanings of words change from time to time. It's hard to decide whether meaning is the right answer when finding synonyms.

With above observations, our future researches will be how to distinguish different word senses from its context features. Once we could distinguish the corresponding features for different senses, it will help us to extract more accurate synonyms for both frequent and infrequent words.

[3] This may due to different genres. In newspapers the proper noun usage of *guang1hua2* is more common than in a literature text.

References

April Kontostathis, William M. Pottenger 2003. , *A Framework for Understanding LSI Performance*, *In the* Proceedings of the ACM SIGIR Workshop on Mathematical/Formal Methods in Information Retrieval, *Annual International SIGIR Conference, 2003.*

Christiance Fellbaum, editor 1998,*WordNet*: An alectronic lwxical database. MIT press, Cambrige MA.

Deerwester, S.,et al. 1990 *Indexing by Latent Semantic Analysis*. Jorunal of the American Society for Information Science, 41(6):391-407

Dominic Widdows. 2003. *Unsupervised methods for developing taxonomies by combining syntactic and statistical information*. In *Proceeding of HLT-NAACL 2003 Main papers*, pp, 197-204.

E.M. Voorhees, "*Implement agglomerative hierarchical clustering algorithm for use in document retrieval"*, Information Processing & Management. , no. 22 pp.46-476,1986

Hofmann, T.1999. *Probabilistic Latent Semantic Indexing*. Proc.of the 22nd International conference on Research and Development in Information Retrieval (SIGIR'99),50-57

James R.Curran and Marc Moens. 2002. I*mprovements in Automatic Thesaurus Extraction. Proceedings of the Workshop of the ACL Special Interest Group on the Lexicon (SIGLEX)*, pp. 59-66

Jia-Ming You and Keh-Jiann Chen, 2004 *Automatic Semantic Role Assignment for a Tree Structure*, Pro-

ceedings of 3rd ACL SIGHAN Workshop

Jiahua Lin. 1991. *Divergence measures based on the Shannon Entropy*. IEEE transactions on Information Theory, 37(1): 145-151

Lillian Lee. 2001. *On the effectiveness of the skew divergence for statistical language analysis*. In Artificial Intelligence and Statistics 2001, page 65-72.

Lillian Lee. 1999. *Measure of distributional similarity*. In Proceeding of the 37th Annual Meeting of the Association for Computational Linguistics (ACL-1999), page 23-32.

Masato Hagiwara, Yasuhiro Ogawa, and Katsuhiko Toyama. 2005. *PLSI Utilization for Automatic Thesaurus Construction. IJCNLP 2005, LNAI 651,* pp. 334-345.

Mei,Jiaju,Yiming Lan, Yunqi Gao, Yongxian Ying (1983) *同義詞詞林 [A Dictionary of Synonyms]*,Shanghai Cishu Chubanshe.

Mochihashi, D., Matsumoto, Y.2002. *Probabilistic Representation of Meanings*. IPSJ SIG Notes Natural Language, 2002-NL-147:77-84.

T.Cover and J.Thomas, 1991. *Element of Information Theory*. Wiley & sons, New York

Appendix:

Some feature clustering results

一下 一陣子

一千多 四百多 一百多 二百多

一切 災難性

一月份 二月份 類型

一生 畢生 一輩子 生平

一年級 國一 大學部

一成 三成 兩成 二成

一百多萬 三百多萬 一千多萬

一段 大半 較多 多一點 空檔 美東 間隔 尖峰 睡眠 美西 需要

一家人 家人 親人

一席之地 優勢

一氧化碳 沼氣 食物 河豚 粉塵

一級 黨務 行庫 原由 二級

一般性 計畫型

一號機 二號機

一銀 二銀 五金

一審 原審 陪審團 審法院

一樓 大會堂 中庭

一舉一動 動向 事項 言行 舉止

一體 中西文

乙級 技術士 廚師 中餐

丁等 甲等 乙等 丙等 中醫師 優等

七人 九人 六人 九人決策

七夕 西洋

七月號 月刊 八月號 雜誌 消息報 週報 二月號

七成 六成 八成 五成 四成 九成

七股 鰲鼓

七美 東引

九孔 草蝦 鰻魚 石斑魚 虱目魚 文蛤 吳郭魚 牡蠣 甲魚 箱網 蝦子 魚蝦 蚵仔 黑鯛 魚群 蝸牛

九份 草嶺 國姓鄉 瑞芳鎮 竹北市

二仁溪 大漢溪 淡水河 漢江 新店溪 游泳池 泳池 鴨綠江 朴子溪 後龍溪 農漁局

二月 十二月 九月 十一月 十月 七月 元月

二年制 四年制

二年級 五年級 大二

二次大戰 第二次世界大戰 大戰 韓戰

二兵 上士 准將 新聞官

二者 兩者 三者

二金 三金 一金

二段 三段

二胡 琵琶 古箏 吉他

二重 疏洪道

二氧化碳 廢氣 污染物 廢水 二氧化硫 氣體 柴油車 氧化物 臭氣 污染源

二專 四技二專 學校院 校院

二組 三組 五組 八組 標準組 組別 梯隊

二號 三號 四號 五號 一號 太原 風雲 型號

二路 三路

二線 三線 四線

二壘 三壘 隊友

二讀會 三讀會 讀會 提案

人人 舉世 舉國 兩性

人力 物力 頻寬

人口 人數 人口數 救濟金 大軍 週數 戶數 家數 次數 隊數 保險金 頻率 斷面

人才 人材 師資 運動選手 增長點 搖籃 英才 專才

人文 美學 藝能 科展

人文組 數理組

人犯 罪犯 煙毒犯

人生 寶島 山城

人生觀 價值觀 觀念

人次 車次

人行道 騎樓

人身 信仰 言論 性行爲

人事費 醫療費用 利息 保費 保險費

人協 黨代表 會員 懇親 股東 社員 締約國

人命 性命 生命

人物 學府 旅遊點 傑作 寶庫 勁旅