Deliverable D2.4.1

Early Informal Language Structure Extraction Prototype

Editor:	
Author(s):	Zhixing Li (THU), Xavier Carreras (UPC), Siqiang Wen(THU), Lluís Padro (UPC), Audi Primadhanty (UPC), Jordi Turmo (UPC)
Deliverable Nature:	Prototype (P)
Dissemination Level: (Confidentiality)	Public (PU)
Contractual Delivery Date:	M21
Actual Delivery Date:	M21
Suggested Readers:	All partners of the XLike project consortium and end-users
Version:	1.0
Keywords:	Linguistic analysis, natural language processing, informal languages,

© XLike consortium 2012 – 2014

Disclaimer

This document contains material, which is the copyright of certain XLike consortium parties, and may not be reproduced or copied without permission.

All XLike consortium parties have agreed to full publication of this document.

The commercial use of any information contained in this document may require a license from the proprietor of that information.

Neither the XLike consortium as a whole, nor a certain party of the XLike consortium warrant that the information contained in this document is capable of use, or that use of the information is free from risk, and accept no liability for loss or damage suffered by any person using this information.

Full Project Title:	Cross-lingual Knowledge Extraction
Short Project Title:	XLike
Number and Title of Work package:	WP2 – Multilingual Linguistic Processing
Document Title:	D2.4.1 – Early informal language structure extraction prototype
Editor (Name, Affiliation)	5
Work package Leader (Name, affiliation)	5
Estimation of PM spent on the deliverable:	

Copyright notice

© 2012-2014 Participants in project XLike

Executive Summary

We have witnessed the rapid development of user-interactive website and applications such as blogs, microblogs and online forums. Emarketer's report [1] estimates that in 2013, social network will cover 25% of the word population and numerous UGCs has been posted or published every day. Nowadays, the amount of UGCs has far exceeded the amount of newswires. Therefore, it is worth paying more attention on extracting knowledge from UGCs.

Since NLP tools developed for standard languages performs much worse on informal languages due to the domain adaption. In this deliverable we developed new tools specialized for informal languages which can to some extent solve this problem. In particular we provide different solutions for Chinese languages and English/Spain languages. In the solution for Chinese UGCs, we developed a new component to discovery of vocabulary words via learning from unlabelled data and another new component to tag these newly discovered words via word clustering. In English, we uses TweetsNLP for shallow processing and then trained a model for named entity recognition. For Spanish, developed a normalization method as a pre-processing modules.

Experimental results show that the proposed solutions can to slightly improve the performance of informal language structure extraction based on limited data. In the future we will improve and test our methods on large scale data.

© XLike consortium 2012 - 2014

Table of Contents

 © XLike consortium 2012 – 2014

List of Tables

© XLike consortium 2012 - 2014

List of Figures

 © XLike consortium 2012 – 2014

Abbreviations

NLP	Natural Language Processing
OOV	Out Of Vocabulary
UGC	User Generated Content
XLike	Cross-Lingual Knowledge Extraction
POS	Part Of Speech
CPOS	Coarse Part Of Speech
MI	Mutual Information
KL-divergence	Kullback–Leibler divergence
KNN	K-Nearest Neighbour

© XLike consortium 2012 - 2014

1 Introduction

It has been proved that the NLP tools developed for formal language meet a significant performance loss on informal language from simple tasks (e.g., PoS tagging) to complex tasks (e.g., dependency parsing) [2, 3]. Structure extraction, which is a task based on these tools, will unavoidably suffer from their performance loss. Therefore, to develop an effective structure extraction component, we need to start with the simplest tasks such as tokenization, lemmatization and PoS tagging.

In D2.3.1, several phenomena in English informal language are discussed. These phenomena are the potential cause of performance loss of NLP tools. I.e., Differences in lexical choice, poor spelling and grammar, difference in prevalent syntactic structure and difference in Discourse structure. Since XLike covers at least seven languages and different languages have different characteristics, the difficulties of informal language processing varies across different languages. For example, In Chinese, tokenization is the very first step of NLP which tries to find the borders of words from a sequence of characters. However, this step is not necessary for almost all western languages where in most cases words are separated by spaces. Consequently, it is not realistic to develop a universal solution which is proper for all languages. In this deliverable, we start by some observations and figures of the phenomena in informal Chinese language and English/Spanish language, and then proposes two different solutions to dealing with these three languages.

 © XLike consortium 2012 – 2014

1.1 Observations on Informal Chinese Language

To find the discrepancies between the informal and formal Chinese language, we conduct several statistical experiments on a corpus which contains 10,000 manually labelled tweets crawled from Sina WeiBo[1], the largest microblogging website in China and CSDN (Chinese semantic dependency network), which contains 100,000 manually labelled sentences from RenMingRiBao, the official newspaper of Chinese government. Table 1 shows the difference between these two copra.

Table 1: The comparison between formal and informal Chinese language

# of words	WeiBo v.s. WeiBo	CSDN v.s. CSDN	WeiBo v.s. CSDN
Top 100	0.006738213	0.043024154	1.308910715
Top 1000	0.032006909	0.128268614	1.388961894
Top 10000	0.571439012	0.390649822	1.760274746
Top 20000	0.826498575	0.503029889	1.874807464
All	0.826498575	0.543025507	1.913957825

Table 1 shows the KL-divergence between the word distribution between WeiBo and CSDN. The first column is number of words. The second column shows the KL-divergence between one 5,000 WeiBo sentences and the other 5,000 WeiBo sentences. The third column shows the KL-divergence between one half of CSDN and another half. The last column shows the KL-divergence between WeiBo and CSDN.

From the figures shown in table 1, it can be observed that there exists a significant difference between WeiBo and CSDN while the divergence inside WeiBo or CSDN is much slighter. This divergence leads to a drop of the word segmentation (see Figure 1).

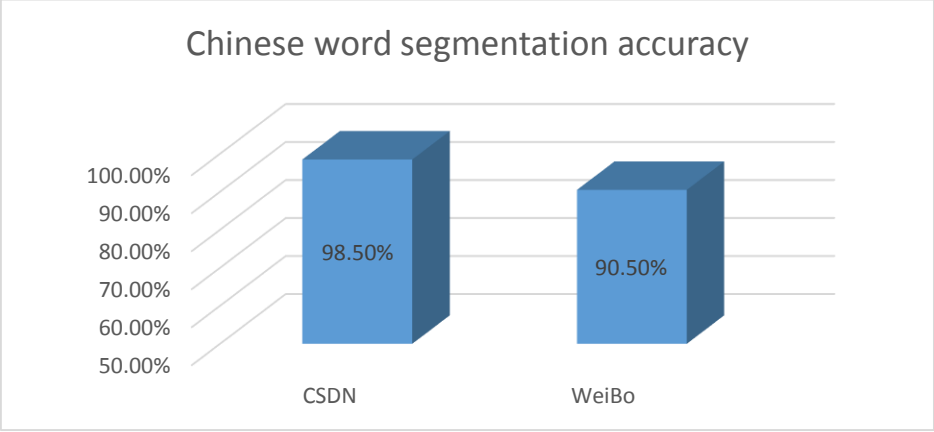

Figure 1: Performance drop of Chinese word segmentation on informal language

[1] http://www.weibo.com

1.2 Observations on Informal English Language

In deliverable D2.3.1 we analysed the statistics of words occurring in different types of corpora in English. In particular we looked at professionally edited news articles from the Wall Street Journal (WSJ), a collection of web blocs, a collection of emails, and a collection of tweets. In the former, standard language is used, while in the latter datasets we expect to find informal language. The experiment takes WSJ as the reference corpus, because most resources used to develop statistical tools for NLP rely on linguistic annotations in these type of datasets. Then we look at the proportion of words in the rest of corpora that do not appear in the WSJ corpus (we reserve a "test" portion of WSJ to count the number of words that do not appear in there, relative to the rest of the WSJ corpus). Table **2** shows the results. As we can see, the proportion of unknown words significantly increases when we go from WSJ, to blogs, email or tweets. It is of capital importance to develop statistical methods for linguistic analysis that can reliably make predictions on sentences where a significant number of words are not in the training portion. This can be achieved using semi-supervised approaches that combine labelled data, with large portions of unlabelled data that contains statistics of all words in the language.

Table 2: Statistics of unknown tokens in datasets of different styles

	WSJ	Web Blogs	Emails	Twitter
Sentences	1,336	1,016	2,450	295,057
Tokens	30,178	22,971	27,739	3,832,108
Unknown Tokens	0.77%	1.31%	2.97%	10.93%
Num/Pun. Tokens	20.85%	16.31%	18.47%	24.69%

© XLike consortium 2012 – 2014

2 Techniques for Structure Extraction on Informal Languages

In the D 2.1.1 and D 2.2.1, a framework for shallow and deep linguistic processing pipeline were proposed. In XLike, we separate the processing of text into several modules, i.e., tokenization/lemmatization, PoS tagging, named entity recognition, dependency parsing, semantic role labelling and then the extraction of relations. Figure 2 gives a brief view of this pipeline.

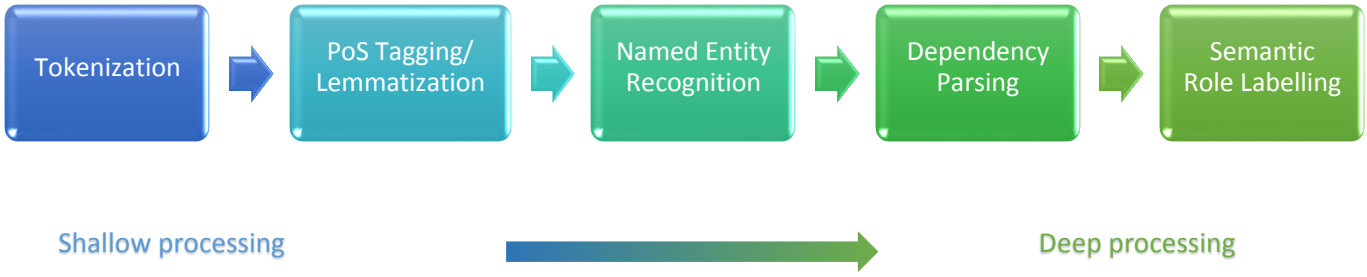

Figure 2 Pipeline of linguistic processing in XLike

It has to be noticed that in XLike, each language has an independent processing pipeline and modifications on one language won't affect the processing other languages. This framework facilitates our prototype which develop different solutions for different languages.

In this section, we give an early informal language structure extraction prototype for Chinese, English and Spanish separately. The prototypes presented in this deliverable are not complete pipeline yet but the some modules or pre-processers which are essential for the informal language processing.

© XLike consortium 2012 - 2014

2.1 Early Prototype for Informal Chinese Language Structure Extraction

For the processing of informal Chinese language, we take a strategy that developing a pre-processing component which aims to improve the performance of word segmentation and POS tagging. We do this based on two considerations.

- Among all XLike languages, Chinese is one of the most "special language" that has no explicit border between words and a single Chinese character rarely has full meaning. Before all operations, we need first identify the border of the words for a Chinese sentence. Usually, a Chinese segmentation tool is trained on a manually segmented corpus and the finding of border is treated as a sequential labelling problem so that it can be modelled using HMM or CRF. A model trained using HMM and CRF is based on the samples which appears relatively frequent in the training data and it usually is unable to find new words which never occurs in training data. In most cases, these words are segmented as single characters. If there is a component that can discover new words from informal languages and automatically label them with correct PoS tags, then the performance of subsequent modules will be improved.
- The second consideration is also related to the characteristics of Chinese language. Chinese is a meaning-cantered/parataxis language. The structure of Chinese is not so strict as English even though in formal text. The meaning of a sentence, is mostly decided by the words it contains instead of its structure. Therefore, it is reasonable to pay more attention on how to correctly recognize the words instead of dealing with different types of sentence structures. What's more, the difference between informal Chinese language and formal Chinese language is not so significant which means the benefit of developing a new dependency parsing module won't be significant.

Based on these reasons, we focus on the recognition and PoS tagging of new words in this early prototype for Chinese. By observing large amount of Chinese tweets, we found that there are five categories of OOV words.

1) Abbreviation. An abbreviation compresses a long words or proper noun into a short word. E.g., "央视" is the abbreviation of "中央电视台(the central television channel)". These are frequently used in oral Chinese and become popular in online social network. There are also some abbreviations originate on the internet. E.g., "高帅富" is short for "高大帅气富有(tall-handsome-rich)". Both these abbreviations are rarely used in formal Chinese language so can hardly be recognized by word segmentation module trained on out-of-date copra.

2) Homophonic. This is one of the most frequently source of new words in online social network. In Chinese, people usually input Chinese characters by their pronunciations, and the same pronunciation may be corresponded with more than one words and some of them causes interesting effects and then become popular. E.g., someone uses "童鞋(children's shoes)" as "同学(classmates)".

3) Isomorphic. Chinese is a kind of pictograph and usually the Chinese characters (or "letters") are composed from more than one stroke or parts. This makes it possible that some of Chinese characters share very similar shapes. Therefore, in online social network, people may use characters in similar shapes. It usually causes no trouble for human reading but can't be recognized directly by machines. "Martian language" is used by some young people. An example of "Martian language" is "吹煋哎" → "火星文". Generally, "Martian language" is used by only a small portion of online users but they are usually loyal to it.

4) Metaphors. On the internet, people may use some metaphors that are not popular on real life. For example, on *Tianya luntan*, which is one of the most popular BBS site in China, users often use "兔子(rabbit)" to represent "中国(China)". Usually, these words can be correctly segmented and tagged but it is difficult for machine to understand the real meaning.

 © XLike consortium 2012 – 2014

5) New words. The cases mentioned above are the informal usages of existing words which are actually not new words. There are also some truly new words that can't be recognized by components trained on out-of-date copra. One source of these words is foreign words. For example, "吐槽(vomit slot)" is imported from Japan and now is very popular on the Chinese online social network, even in daily life. Another source is the occurrence or invention of new things. For example, "切糕" is the name of a kind of food which were rarely known by people 5 years ago. These words are in fact not informal words, but they can be recognized using the same technique we used to dealing with the informal usages we mentioned above.

We can divide the processing of OOV words into three steps: I) recognition, 2) tagging and 3 understanding. The first step refers to correctly finding them in character sequences and the second step refers to tagging them with syntax labels such as PoS tagging. The last step is to understand their semantic. In this prototype we deal with the first and the second steps.

2.1.1 OOV Words Recognition

Using a word segment tool trained on formal language will unavoidably split most of the OOV words into isolated characters. For example, a sentence "奥巴马是美国总统(Obama is the president of USA)" will be segmented as "奥/巴/马/是/美国/总统" while the correct segmentation should be "奥巴马/是/美国/总统". Generally, there are two kinds of segmentation method: generative segmentation and discriminative segmentation [4, 5]. The discriminative segmentation algorithms is said to be more powerful in OOV recognition. However, the divergence between formal Chinese and informal Chinese language is too significant to be handled by algorithms and we need to turn to the "Big data" for possible solutions. In this prototype, we propose a new method which tries to find OOV words from large scale unlabelled data under the help of a model trained on out-of-date labelled corpus. The framework of our approach is shown in Figure 3

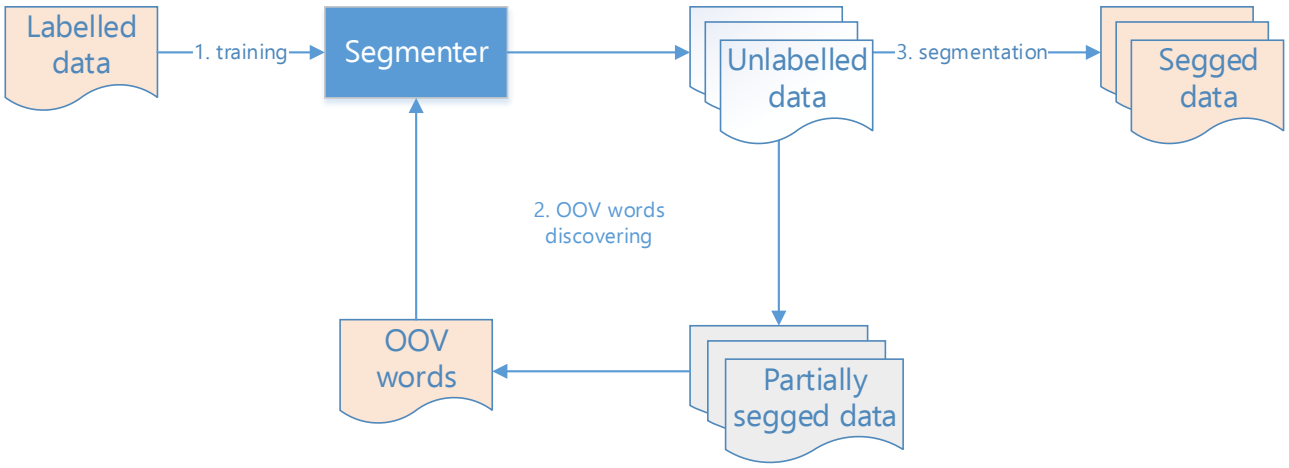

Figure 3 Discovering OOV words from unlabelled data

While the labelled data is expensive and usually of limited scale and can't update as the growing of texts, unlabelled data is to obtain with a relatively very low cost. In our solution, we use both labelled data and unlabelled data.

The basic idea of learning from unlabelled data is to find the patterns which frequently occur in large scale of real world data. However, not all frequently occurs patterns are words. For example, "我是 (I am)" occurs frequently but it should not be recognized as a word. In this prototype, the discovering of OOV words consists of two steps: 1) frequent pattern extraction and 2) non-words pattern filtering.

1) Frequent pattern extraction

The idea behind our solution is that if we want to combine two characters (words) into a new word, these two characters (words) should co-occur in real world data frequently. This goal can be achieved by finding the frequent pattern in copra. Considering some OOV words consist of three or more characters (words), we leverage a recursive manner to find OOV words. In each iteration, only a few OOV words are accepted and then added add to the segmenter as learned knowledge. For example, if we want to discover 200 OOV words from the unlabelled data, we will run the second step for 10 times and in each iteration, only the top 20 OOV words are accept.

To decide if a pattern is frequent, we test three scoring functions:

1) Mutual Information

$$Score_{MI}(w_1, w_2) = p(w_1, w_2)\log(\frac{p(w_1, w_2)}{p(w_1)p(w_2)})$$

2) Custom Measuring 1

$$Score_{CM1}(w_1, w_2) = p(w_2|w_1)p(w_1|w_2) = \frac{p(w_1, w_2)^2}{p(w_1)p(w_2)}$$

3) Custom Measuring 2

$$Score_{CM2}(w_1, w_2) = \log\big(p(w_1, w_2)\big) * Score_{CM1}(w_1, w_2)$$

In all these equations, w stands for a character or a word.

The first scoring uses mutual information between w_1 and w_2. According to information theory, MI a quantity that measures the mutual dependence of the two random variables. Higher MI value suggests stronger dependency between the two random variables. Strictly speaking, $Score_{MI}$ is not the MI between w_1 and w_2 because both w_1 and w_2 are samples instead of random variables.

The second scoring uses the product of two conditional probabilities. We avoid to use a single conditional probability to reduce the impact of high-frequent characters (words).

The third scoring is derived from the second one. The only change is timing a log factor to give more value to frequent pairs in case $p(w_1)$ or $p(w_2)$ is too large.

2) Non-word pattern filtering

As discussed, not all frequent patterns are words. Therefore, after the extraction of frequent patterns, we need to filter the non-word patterns out. This task is essential for the precision of extracted new words. In this prototype, we use a background corpus to filter non-word patterns. We suppose that an OOV word in informal language should not occurs in out-of-date formal language because usually it is invented or used recently. If a frequent pattern we extracted from informal language is also frequently occurs in out-of-date formal language, then it is probably a syntactic pattern instead of a word.

Datasets and Experiments

We test our solution on two datasets. The first one is a small dataset collected from Weibo contains 10,000 sentences. These sentences are manually segmented. The second dataset is much larger than the first one but hasn't been manually processed. We use the first dataset to evaluate the performance gain after OOV words recognition and use the second dataset to evaluate the precision of OOV discovering algorithm. We cannot test the recall because we have no idea about the total number of OOV words. The datasets are illustrated in Table **3**:

 © XLike consortium 2012 – 2014

Table 3: Illustration of datasets for OOV discovering

Dataset	# of tweets	# of sentence	# of characters	Evaluation metrics
Weibo1	--	10.000	184.964	Word segmentation accuracy
Weibo2	4.851.000	--	254.013.471	Precision of extracted OOV words

Weibo1 is manually segmented, tagged and labelled with dependencies.

The evaluation metrics is calculated as follows.

Word segmentation accuracy:

$$Acc = \frac{\#\ of\ correctly\ segged\ words}{\#\ of\ words}$$

Precision of OOV words:

$$Pre = \frac{\#\ of\ correctly\ extracted\ OOV\ words}{\#\ of\ extracted\ OOV\ words}$$

For the first dataset, we extract 1.000 OOV words in 50 iteration and the experimental result are shown in Figure 4.

We can see that the segmentation accuracy is obviously improved after OOV discovering no matter which scoring methods is used. Not surprisingly, the performance curve of these three methods are different with each other. During these three scoring methods, MI reaches the best performance when only a few OOV words are used and it means MI method ranks the most popular OOV words high but it gives more priority to frequency rather than co-relation between characters (words) so that the performance reaches the peak soon and after that, when number of extracted OOV words increases, the performance decreases. The CM1 method is another case. The performance gain of CM1 method is not significant at the beginning but lasts for a long period as the increasing of extracted OOV words. It means the high rank OOV words of CM1 method are not of high frequency but are truly new words. CM2 is derived from CM1 by adding a frequency factor $p(w_1, w_2)$ which will slightly increase the rank of frequent patterns. Therefore, we can observe a rapid and continuous performance gain when CM2 scoring method is used.

© XLike consortium 2012 - 2014

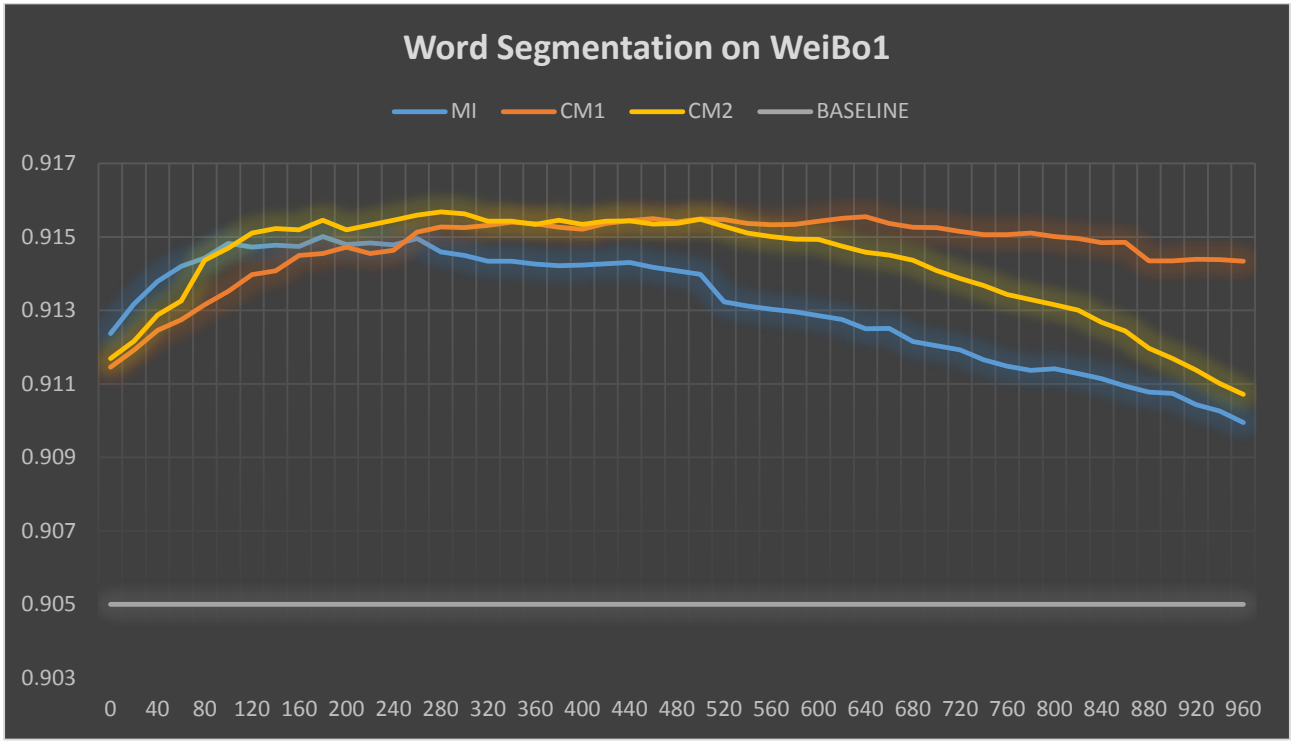

Figure 4: Word Segmentation after OOV words extraction.

After the test on WeiBo1, we found that CM2 is the best method to extract OOVs from informal language so in the second test, we use CM2 to extract OOV words from WeiBo2, an unlabelled corpus. WeiBo2 is a large corpus and we extracted a lots of OOV words from it (over 20,000). Since we cannot check these OOVs one by one, we just checked the top 100 ranked words and another 500 randomly selected from top 2000 words. Finally, the precision of OOV word extraction on WeiBo2 is illustrated at Figure 5.

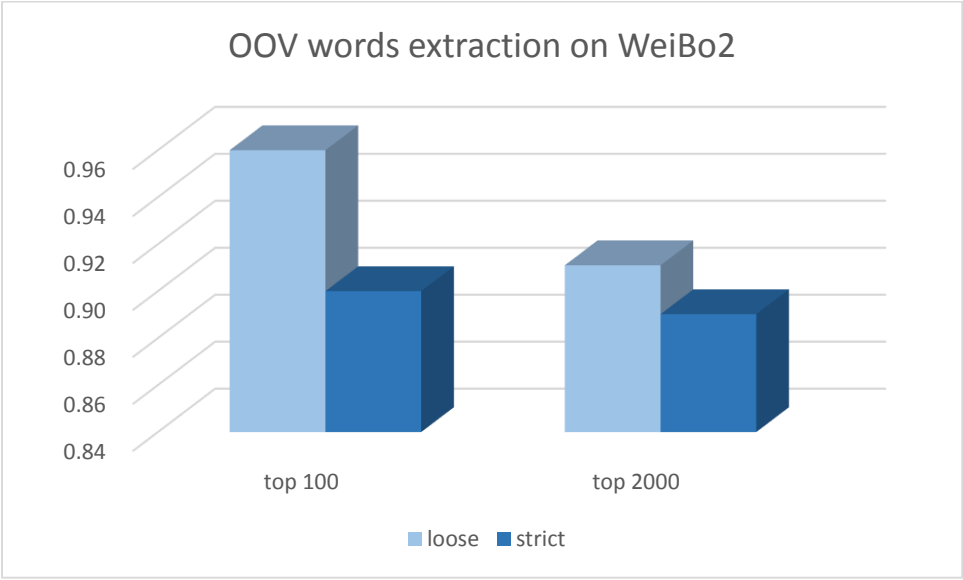

Figure 5: OOV words extraction on WeiBo2.

In Figure **5**, strict means that the extracted word should be a complete word and loose means that the extracted words is either a whole word or a part of a words. The fragments of words may be merged into a complete words in subsequent iterations. Using the loose standard, the precision of top 100 is about 96% which is comparable with the performance of word segmentation on formal language. However, the precision drops to about 91.1% on top 2000 extracted words. The precision under strict standard keeps

 © XLike consortium 2012 – 2014

almost the same (about 90%). In the future, we need to find new methods to exclude non-words patterns to improve the precision of OOV words extraction.

2.1.2 OOV Words PoS Tagging

After the recognition of OOV words, in this subsection we propose a method for the PoS tagging of these extracted OOV words. PoS tagging is essential for following modules such as named entity recognition and dependency parsing. Because we have no knowledge about the OOV words, we cannot give them the tags directly. In this prototype we use KNN algorithm to tag OOV words. For a sentence that contains some OOV words, we first tag all words but the OOV words using existing PoS tagging tools. Then we transform each word (including the OOV words and normal words) into a vector and leverage a classifier trained on normal words to tag the OOV words. To transform a word into a vector, we used 1) the Form, PoS tag, CPoS tag of the previous word and 2) the Form, PoS tag, CPoS tag of the previous word. Then the tagging task is cast into a classifying problem and solved using KNN algorithm.

We test our method on WeiBo1 dataset where we have manually tagged words. The experimental results are shown in:

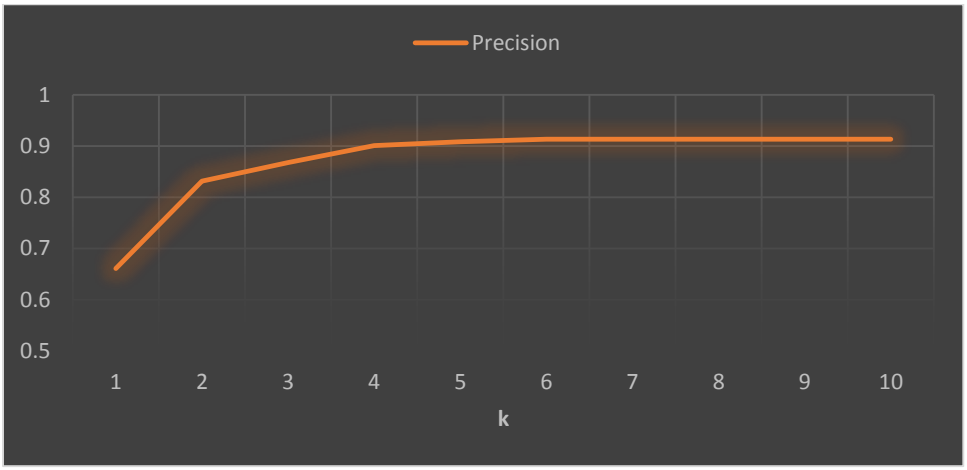

Figure 6: Chinese OOV tagging precision of top K predict tags

Figure 6 shows the OOV words tagging precision in informal Chinese language. The precision of the top 1 predict tag is about 66% which is much lower than the performance on formal language. The precision of the top 2 predict tag is about 83%. The cause of the low performance is on two-fold. On one hand, in this prototype, we used only the neighbouring information in feature set and no sentence level syntax information is used. On the other hand, the lack of labelled data hampered the tagging task. In some extreme (but not rare) case, the OOV words to be tagged occurs only one time in the labelled corpus so that it very difficult to do the correct prediction.

© XLike consortium 2012 - 2014

2.2 Early Prototype for Informal English Language Structure Extraction

Our first prototype for processing informal English tackles the shallow components of the pipeline. In particular we have assembled a new pipeline for informal English, targeted at tweets, consisting of:

- Tokenization and PoS tagging, using the TweetNLP tools from Carnegie Mellon University [6].
- A named entity recognition system (NER), developed by UPC, based on Conditional Random Fields (CRF) and following the ideas in [7] and [8].

The NER system corresponds to a standard CRF-based method for predicting named entities using sequential information. The features include standard features for NER, namely ortographic patterns of words; lexical features including bigrams; and gazetteers. We used the datasets released by [8], that annotate English tweets using a taxonomy of 10 types of tweets (person, location, organization, company, facility, movie, music artist, product, sports team, TV show, and other). We followed a 10-fold cross-validation strategy in order to evaluate the CRF model. Table 4 shows the results in terms of precision, recall and F1 measure. As we can see, the overall performance is of approximately 48 in F1, which is quite lower than the performance in standard texts (close to 90%). If we look at the performance of segmenting named entities (without predicting their type), the performance is at 55 in F1; hence most of the errors are already in identifying named entities. This may be due to the fact that capitalization information is not reliable in tweets, but is very helpful in standard texts in order to detect entities. The table also shows the breakdown of performance with respect to the type of entity. We can see that the performance at recognizing persons, locations and organizations is in the range of 60 in F1, which is significantly better than the overall performance. On the other hand, recognizing products or music artists has very low performance.

Table 4: Performance of a CRF-based Named Entity Recognizer on English Tweets

	Precision	Recall	F1
Segmentation and Classification	61.97	39.15	47.92
Segmentation only	72.04	45.49	55.69
Person, Location, Organization	65.99	55.53	60.26
Company	64.75	41.39	49.88
Facility	45.33	15.95	22.31
Geo-Location	67.40	55.41	59.62
Movie	40.00	15.54	19.17
Music Artists	21.67	8.08	11.62
Person	65.31	60.84	62.74
Product	25.00	3.73	6.33
Sports Team	55.00	16.93	25.27
TV Show	45.00	15.10	21.67
Other	44.24	18.49	25.20

© XLike consortium 2012 – 2014

2.3 A Method for Normalizing Spanish Tweets

We now describe a system developed at UPC to normalize OOV words occurring in Spanish tweets. The system was developed in the context of the TweetNorm shared task, organized by the SEPLN society [9].

2.3.1 Description of the System

The UPC system for SEPLN 2013 Tweet-Norm shared task [10] consists of a collection of expert modules, each of which proposes corrections for out-of-vocabulary (OOV) words. The final decision is taken by weighted voting according to the accuracy of each expert on the development corpus.

First, a preprocessing step is applied, where consecutive occurrences of the same letter are reduced to one (except valid Spanish digraphs like *rr* or *ll*). We generate also a version of the OOV with those repetitions reduced to two occurrences (to capture cases such as *coordinar*, *leed*, *accin*}, etc.). In this way, we obtain three different OOV versions (original, reduction to one repeated letter, reduction to two repeated letters) that will be checked against dictionaries and gazetteers as described below.

All expert modules are implemented using FreeLing library facilities for dictionary access, multiword detection, or PoS tagging. Some experts use string edit distance (SED) measures to find words in a dictionary similar to the target OOV. FOMA library [??] is used in these cases for fast retrieval of candidates.

The used expert modules can be divided in three classes:

- Regular-expression experts: Experts in this class are regular expression collections that propose corrections for recurring patterns or words, such as smileys, laughs (e.g. jajjaja, jeje, etc.), frequent abbreviations (e.g. `TQM -> te_quiero_mucho`, `xq -> porque`, etc.), or frequent mistakes (e.g. `nose -> no_sé`). Experts in this category propose a fixed solution for each case.
- Single-word experts: Each module belonging to this class uses a specific single-word lexical resource and a set of string edit distance (SED) measures to find candidates similar to the target word. The three SED measures specifically used for the task are: character distance (the conventional edit distance metric between strings), phonetic distance (transformations according to similarity in pronunciation) and keyboard distance (transformations due to possible errors when typing).
- Multi-word experts: Modules in this category take into account the context where an OOV is located to select the best candidate among those proposed by the other experts. We used three different experts in this category. First, the *multiword dictionary* module takes into account proposals of the single-word experts that use different distances over a dictionary consisting only of tokens that appear in known multiwords. All combinations of possible candidates for the OOV and its context are checked against the multiwords dictionary, and those matching an entry are suggested as corrections. Second, the PoS tagger expert takes into account all proposals of all single-word experts, retrieves the possible PoS tags for each of them, and creates a virtual token with a morphological ambiguity class including all obtained categories. Then, a PoS tagger is applied, and the best category for each OOV is selected. The module filters out all proposals not matching the resulting tag, and produces as candidates only those with the selected category. Finally, the *glued words* expert, which consists of a FSM that recognizes the language *L(_L)+*, where *L* is the language of all valid words in the Spanish dictionary. Using foma-based SED search on this FSM with an appropriate cost matrix, we can obtain, for instance, that `lo_siento` is the word in the FSM language closer to `losiento`, and propose it as a candidate correction.

After all experts have been applied, a selection function is used on the set of resulting candidates. This selection function takes into account the SED distance of each proposal to the original OOV, the number of

experts that proposed it, and the precision, recall, and F1 of each expert on the development corpus to perform a weighted voting and select the final correction.

The system makes use of several resources, namely:

- Gazeteers of acronyms, emoticons, and onomatopoeias.
- Dictionaries and gazeteers for single-word experts. These include Spanish and English dictionaries, as well as common named entity names.
- Gazetteers of multi-word named entities.

See [10] for a detailed description of the resources and how they were obtained.

2.3.2 Experiments and Results

Our system has a total of 32 different producers that are integrated in our tweet normalizer. Additionally, we add a 33rd producer that always proposes to leave the target OOV as it is. The combined outputs of these producers yield several hundreds of spelling alternatives for the OOV words, therefore we need a principled method to choose the best one among them, including to leave the original word as it is given. This strategy is able to propose the correct spelling alternative to 89.42% of the OOVs found in the development corpus, therefore, this is the upper-bound accuracy of our system.

Using the development corpus, we have computed the precision, recall and F1¡ of each producer. Since the producers yield a list of spelling alternatives that are sortable according to the SED metrics, we have devised three different levels where we can measure its confidence:

- TopN: At this level, we check only if the producer produces the correct correction anywhere in the alternatives list.
- Top1: This level checks how many times the correct correction has the smallest SED in the whole list of alternatives (i.e., it is on the front of the list). In this case, the precision is computed against the total number of proposed corrections having the smallest SED.
- Top0: This measures how many times the correct correction has a SED distance of zero over the total number of proposals at distance zero. Note that all exact searches (regular-expression experts and look up dictionaries) yield alternatives with distance zero.

We compute precision, recall and F1 for each producer for all three levels of measure.

To produce a proposal for each OOV, we implement a voting scheme. Each producer votes for each of their proposed corrections using the suitable TopN, Top1 or Top0 scores as their vote weight. The possible corrections are pooled together and the one with the largest total score is our final proposal. Note that a proposed correction in the Top0 position is also in the Top1 and the TopN positions. Therefore, we can choose if the weight of a producer vote is just the score of it's best measure (e.g. Top1 instead of TopN) or the addition of all suitable measures (e.g. Top1 plus TopN for a proposal in Top1). We have experimented with these two voting schemes that we call *single* or *additive* and with using precision (P), recall (R) or F1 as the actual vote weight. We have also considered the possibility of squaring the weights in order to strengthen the relevance of high precision producers.

 © XLike consortium 2012 – 2014

Table 5: Evaluation of alternative voting schemes for tweet normalization with experts

Weight	Scheme	Normal	Squared
R	Single	54.79	65.92
R	Additive	60.77	69.12
P	Single	65.78	67.45
P	Additive	67.45	67.81
F1	Single	65.09	67.87
F1	Additive	67.87	69.12
W100	Single	59.52	---
W110	Single	41.16	---
W111	Single	23.78	---
W111	Additive	45.61	---

Table **5** shows the results achieved using the development corpus for testing and estimating the weights. We have set up some baselines giving fixed weight to the votes. We have the scheme *w111*, which gives a weight of 1 to TopN, Top1 and Top0; the scheme *w110* gives a weight of 0 to TopN and of 1 to the other two, and so on. The experiments show that using squared precision as the confidence measure within an additive scheme yields the best results: a precision of 69.81% on the development corpus.

The official result of our single run is an accuracy of 65.26% on a test corpus of 500 tweets containing roughly 700 OOVs. In this run we use the weights estimated from the development corpus. This results is 4.5 points behind what we obtained on the development corpus, suggesting that our estimation method is reasonable but may be overfitting.

To elucidate this issue, we have repeated our experiment using the test gold standard to estimate the vote weights (instead of using the development data). With this setup and identical voting scheme, the precision is increased by 0.76 points, less than four points behind the 69.81% we got in the development set. Additionally, we have used the gold standard to calculate the upper-bound of our producers as we did with the development data. We are able to propose the correct word for 85.47% of the OOVs, which is 4 points behind the 89.42% for development data.

Since little improvement is obtained when using the test data, this suggests that our strategy of estimating each producers' precision is not overfitting. Additionally, we can see how the drop in the system's upper-bound matches its accuracy drop. Therefore, we believe that the nature and distribution of OOVs in Twitter streams may vary over time more than it is represented on the development set, thus, our strategy as a whole is more suited to this particular set of development data than to the test data.

2.3.3 Future Lines

Our approach achieved a precision of 65.26% in the test corpus of TweetNorm 2013 evaluation, ranking our system in the 3rd best place among the participants. This result shows the appropriateness of our approach for the task. However, it is far to achieve the upper-bound results (i.e., from the 69.81% achieved for the development corpus to the upper-bound of 85.47% achievable in that corpus). This fact shows that there is room enough to improve our system.

© XLike consortium 2012 - 2014

In order to get improvement, main lines in our future work involve enriching the lexical resources with OOV words occurring in the unannotated tweets provided by the organizers, using a richer context of the OOV words to drop out false candidates, tuning the costs of the edit distances operators, and considering other alternative voting schemes.

© XLike consortium 2012 – 2014

3 Conclusion

In this deliverable, we have presented an early prototype for Chinese OOV words recognition and PoS tagging, an early prototype for informal English language structure extraction, and a method for the normalizing of Spanish OOV tweets. The data observations shows that informal language differs from formal languages in a significant level on both Chinese and English and these differences do cause performance drops dramatically.

In Chinese OOV recognition, we used a statistical method to find the frequent pattern in informal Chinese language data to discover potential OOV words. We tested three scoring methods for the patterns. However, the patterns may be cause by not only popular OOVs but also frequent syntax structures. To overcome this, we filtered the extracted patterns using background corpus containing formal Chinese texts, which is supposed has similar syntax structures but no OOVs. This method works well to some extent and in the future we will find some other methods to identify if a pattern really leads to OOV words.

The prototype for informal English language also sees a performance drop comparing with what for formal English. The main reason is the capitalize information, which is important for the identifying of named entities, is not reliable in informal language data such as Tweets.

In this deliverable we also presented a normalization method for Spanish tweets which makes use of different kinds of resources. The performance is satisfying as for an early prototype while there is still lots of room for improvements. In the future will exploit more resources to drop false candidates.

Combining all these three approach, we find that main challenge in the word/token level is how to improve the precision of our methods while in the named entity level, improving the recall is more challenging. Considering there is actually no algorithm or method is special for informal language processing, a reasonable direction in future work is to exploit more recourses and find more features in informal language processing.

© XLike consortium 2012 - 2014

4 References

[1] Worldwide Social Network Users: 2013 Forecast and Comparative Estimates, http://www.emarketer.com/corporate/reports, *2013*

[2] XLike Deliverable D2.1.1 --- Shallow linguistic processing prototype.

[3] Petrov S., McDonald R., *"Overview of the 2012 Shared Task on Parsing the Web", in SANCL 2012.*

[4] Hai Zhao, Chang-Ning Huang, Mu Li, and Bao-Liang Lu. 2010. A Unified Character-Based Tagging Framework for Chinese Word Segmentation. 9, 2, Article 5 (June 2010), 32 pages.

[5] Kun Wang, Chengqing Zong, and Keh-Yih Su. 2012. Integrating Generative and Discriminative Character-Based Models for Chinese Word Segmentation. 11, 2, Article 7 (June 2012), 41 pages.

[6] Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. Part-of-Speech Tagging for Twitter: Annotation, Features, and Experiments. In *Proceedings of ACL 2011*. http://www.ark.cs.cmu.edu/TweetNLP.

[7] Xiaohua Liu, Shaodian Zhang, Furu Wei and Ming Zhou. Recognizing Named Entities in Tweets. Proceedings of ACL 2011, pages 359-367.

[8] Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. Named Entity Recognition in Tweets: An Experimental Study. In Proceedings of EMNLP 2011.

[9] Tweet Normalization Workshop at SEPLN 2013, http://komunitatea.elhuyar.org/tweet-norm.

[10] Alicia Ageno, Pere R. Comas, Lluís Padró and Jordi Turmo. The TALP-UPC approach to Tweet-Norm 2013. Tweet-Norm workshop, SEPLN 2013.

[11] Mans Hulden. Fast approximate string matching with finite automata. Procesamiento del Lenguaje Natural, 43, pages 57-64. 2009.

 © XLike consortium 2012 – 2014