Improvement Recall of NTCIR-MedNLP using Hierarchical Bayesian Language Models

Ryo Fujii
Keio University, Japan
roy@ae.keio.ac.jp

Masashi Tada
Mathematical Systems, Inc., Japan
tada@msi.co.jp

ABSTRACT

The msiknowledge team participated in the de-identification, complaint and diagnosis subtasks of the NTCIR-10 MedNLP Pilot Task. The terms of complaint and diagnosis subtasks may have several representations and some terms may be not in sample data, and also locations and personal names in de-identification task are unique in each cases, so simple dictionary building from sample can't solve the problem.

In this research, to avoid using outer copora or dictionaries which are not general solution, we aimed at building a system that learns to recognize terminology from small tagged corpus by combining a tag level language model and a character level language model base on HPYLM.

Team Name

msiknowledge

Subtasks

MedNLP de-identification task
MedNLP complaint and diagnosis task

Keywords

POS tagging, Hierarchical Bayes, Language Models

1. INTRODUCTION

The msiknowledge team participated in de-identification, complaint and diagnosis subtasks of the NTCIR-10 MedNLP Pilot Task[5]. The terms of complaint and diagnosis subtasks may have several representations and some terms may be not in sample data. More generally, new technical terms are invented on a daily basis, thus it is impossible to deal with such new technical terms only by simply preparing dictionaries in real applications. Also locations and personal names in de-identification task are unique in each cases, so simple dictionary building from sample can't solve the problem.

But we aimed avoiding to use outer copora or dictionaries, because these solutions are costly to compile them and not effective in real applications as mentioned above. Instead of these, we aimed at building a system that learns to recognize terminology from small tagged corpus which may be useful for other areas.

In this paper, we used two types of **Variable-order Hierarchical Pitman-Yor Language Model** (VPYLM)[3], one of which is for tag sequences that will find an appropriate context to recognize common words in a technical term, and the other is for character sequences that will find useful character strings (prefixes or suffixes) to recognize unknown technical terms. To reach higher score, we combined this model and standard conditional random fields (CRF) for submission.

2. HPYLM AND VPYLM

The Hierarchical Pitman-Yor Language Model[6] (HPYLM) is an n-gram language model based on Pitman-Yor language process.

In HPYLM, the depth of the hierarchy is common and fixed for all the contexts. As a result, where n becomes greater, the size of the CRP tree of HPYLM becomes greater in the order of Σ^n . This fact has virtually made the computation of a high dimensional n-gram HPYLM impossible. VPYLM[3] eases this restriction by considering one more probability process on branches.

To explain VPYLM we use a Chinese restaurant process. When a new customer comes to a VPYLM, the customer goes down as in HPYLM but may land at the node earlier. Where to stop is determined according to $p(v)$ defined below.

The probability of passing the node $q(h)$ and that of arriving at the node $1 - q(h)$ can naturally be estimated from the number of passes of each node $t(h)$ and the number of arrivals $s(h)$. With the assumption that all the number of passes of the node and the number of arrivals at the node have as a prior distribution the beta distribution, the above mentioned probabilities are estimated as $q(h) = \frac{t(h)+\alpha_h}{s(h)+t(h)+\alpha_h+\beta_h}$. Therefore the probability $p(h_l)$ that w is observed at the node h is given by $p(h_l) = (1 - q(h_l)) \prod_{i=0}^{l-1} q(h_i)$ where h_l is a lower level context or subcontext of h with the length $l(\leq L)$, where the entire length of h is L.

In addition, by introducing a certain small probability threshold ϵ, we could stop the growth of tree, i.e., stop branching, at the node such that $\epsilon > p(h_l)$ by forcing the customer to arrive at the node. By doing so, we could control the growth of the CRP tree to obtain a reasonable size tree with good accuracy.

3. COMBINING LANGUAGE MODELS

We consider the problem of technical term recognition in the framework POS tagging. We suppose that a technical term has a special POS tag, which is the same as the sample data of MedNLP Pilot Task. Under the framework, we could use context which is a sequence of tags before a target tag to properly recognize compound words.

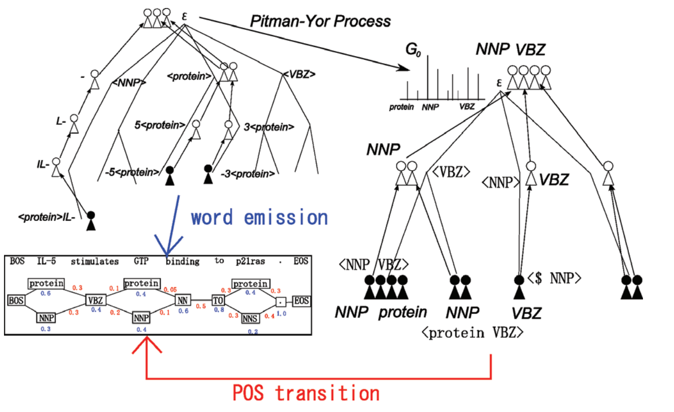

Figure 1: Nested Tag-HPYLM

We think technical terms have their features in their spelling and are reliably predicted based on the features. To incorporate the knowledge of features by machine learning, we propose to build a character-level HPYLM which predicts the probability for word to have a POS or to be a technical term based on their spellings. By combining the tag-HPYLM and the character-level HPYLM, for predicting tag-sequence probability and for predicting word emission probability would get better prediction of probability. The combined model is to be referred to as **nested tag-HPYLM** (ntH). The character-level HPYLM was proposed, combined with word-level HPLYM, and is called NPYLM in [4].

The difference of our proposal with NPYLM is that we introduce some specific tokens in character-level HPYLM, we use VPYLM for tag-HPYLM, and consequently we need to invent new forward filtering-backward sampling scheme.

3.1 Character-level HPYLM

Since we could not foresee the appropriate length of characters to be used for character-level HPYLM and the length could be long (maybe longer than five characters), we use VPYLM instead of HPYLM as in [4].

3.2 Introduction of Tokens

The purpose of character-level VPYLM is to automatically learn characteristic spelling specific in technical terms. The specific spellings are mainly observed in prefixes and suffixes and we need to somehow specify some words are technical terms and some words are not. To solve the problem, we propose to insert some tokens specifying POS or that the following word is a specific technical term and that the preceding word is a specific one.

A token representing "protein" is inserted before and after every word or word strings that mean protein. Now word emission probability from tag t $P_{\mathrm{emit}}(w|t)$ is replaced by $Q(w_t)$, where w_t is $\langle t, c_1, \ldots, c_L, t\rangle$. Assume "IL-5" is a protein name and our character-level VPYLM will learn that the trigram $\langle protein, \mathrm{I}, \mathrm{L}\rangle$ is a frequent trigram. When the VPYLM is asked $Q(\langle protein, \mathrm{I}, \mathrm{L}, -, 2\rangle)$, it would return far larger probability to $\langle protein, \mathrm{I}, \mathrm{L}, -, 2\rangle$ than to $\langle noun, \mathrm{I}, \mathrm{L}, -, 2\rangle$. Then we would say that "IL-2" would be a protein name rather than a commonly used name.

Additinally, the base distribution in POS-VPYLM is estimated from character-VPYLM, For character-VPYLM, every string starts with a POS token, so distribution after "beginning of string" is equal to tag unigram distribution.

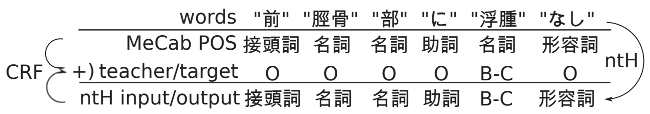
words "前" "脛骨" "部" "に" "浮腫" "なし"
MeCab POS 接頭詞 名詞 名詞 助詞 名詞 形容詞
CRF
+) teacher/target O O O O B-C O
ntH input/output 接頭詞 名詞 名詞 助詞 B-C 形容詞
ntH

Figure 2: ntH+CRF outline

4. INTEGRATION OF NTH AND CRF

Through preliminary experimentation (see detail in Table 2) we found that the ntH models have better recall but worse precision than the baseline (standard CRF). Most popular mistakes of ntH is false detection of "主な入院時現症" as complaint. This word is very popular in the section title of a carte, but it has the suffix "症" so it seems complaints for ntH. We guessed that POS-VPYLM works fine to recognize features of the target terms but whole it can't learn global best tagging results, so we aimed to integrate ntH and standard CRF.

In this research we simply add new feature "output of ntH" to the CRF and call it "ntH+CRF". Figure 2 shows the outline of ntH+CRF. The fourth line, ntH input and output is calculated from the second line, the POS tags and the third line, the teacher signals from datasets. The ntH learn and predict mapping from the words to the ntH sequence, or from the first line to the fourth line. Then CRF learn and predict from the POS sequence and ntH outputs to the target signals, or the second and the fourth line to the third line.

There is an issue in this feature in supervised dataset, because these are supervised signal or correct answer of ntH model. ntHs tries to learn them but there should be some prediction error between the models and supervised data. So when we want to use model outputs for CRF features, we might have to update supervised dataset by ntH output to keep consistency of these features. In other words, there is a trivial mapping from input of ntH to target signal but this mapping is not true in test data because ntH might output wrong features.

But we also suspect the features which contains wrong signal in supervised dataset is useless or just noise. So we compared these two settings;

ntH+CRF original ntH input is used for CRF features in supervised data

ntH+CRF-update ntH output is used for CRF features in supervised data

We implemented ntH and the small script for converting outputs of ntH for CRF using Common Lisp, then we used CRF++[1] for CRF implementation.

5. EXPERIMENTS

5.1 Pretreatments

We segmented learning data and added POS tags by MeCab[2] except technical terms, trained the proposed models.

[1]http://code.google.com/p/crfpp/

Except when comparing CRF-baseline and ntH, we tried to detect the attributes of complaint and diagnosis, "family", "negation", "suspicion" and no attributes. So we separeted complaint and diagnosis into 4 subclassed by its attributes.

To emphasize the prefix/suffix of technical terms for ntH models, we defined a trivial mapping from MedNLP annotation to four kinds of tags, "L-/R-" for left/right boundary, "I-" for middle words, and "S-" for just a single word expression of a term. We used this "L-/R-/I-/S-" expression for input of ntH models. The outputs of ntH models are mapped to CoNLL-2000 shared task format[1], and used for evalution or input for ntH+CRF.

For the evaluation of preliminary experiments within sample data we performed 10-fold cross validation and collect outputs, then used the evaluation script "conlleval" distributed at CoNLL-2000 web site[2].

5.2 CRF features set

We used MeCab outputs and ntH outputs as CRF features.

First we picked

- word expressions
- large POS classes
- small POS classes

from MeCab outputs and decied to use them within two words from the position where feature functions are called. We also defined its combinations of adjacents and combination between different classes in the same position. We call this features set as "MeCab features".

Next we defined ntH outputs without combinations as "ntH simple features" and combinations as "ntH combinatorial features".

With these three sets, we made experiment settings for CRF.

CRF-baseline MeCab features only

ntH+CRF simple MeCab features + ntH simple faetures

ntH+CRF all MeCab features + ntH simple features + ntH combinatorial features (all features)

ntH+CRF-up all all features, ntH output is used for CRF features in supervised data

5.3 Preliminary experiment

Followings shows the evaluation measures of 10-fold cross validation in sample dataset. Frist, Table 1 shows the scores of CRF-baseline and Table 2 shows the scores of ntH.

As previously mentioned, ntH has better recall but worse precision than CRF-baseline in de-identification subtask.

Next 3 tables show ntH+CRF results with settings described in section 5.2.

Thorugh these preliminary experiments, we found ntH+CRF is a little better than simple CRF and ntH. We couldn't find any effect of combination features for accuracies in these subtasks, and updating supervised data to keep consistency looks bad strategy in this data.

[2]`http://www.cnts.ua.ac.be/conll2000/chunking/output.html`

Table 1: Evaluation of CRF-baseline

Tag	Precision	Recall	FB1
A	0.8571	0.6545	0.7423
H	1.0000	0.8493	0.9185
L	0	0	0
T	0.8971	0.8206	0.8571
X	1.0000	0.5000	0.6667
C	0.8732	0.7724	0.8197
Total	0.8804	0.7779	0.8260
Accuracy	—-	—–	0.9639

Table 2: Evaluation of ntH

Tag	Precision	Recall	FB1
A	0.7627	0.8182	0.7895
H	0.8462	0.8800	0.8627
L	0	0	0
T	0.8192	0.8423	0.8306
X	0.2308	0.7500	0.3529
C	0.7250	0.7534	0.7389
Total	0.7408	0.7719	0.7561
Accuracy	—-	—–	0.9500

Table 3: Evaluation of ntH+CRF simple

Tag	Precision	Recall	FB1
A	0.8810	0.6727	0.7629
H	1.0000	0.8219	0.9023
L	0	0	0
T	0.8939	0.8176	0.8541
X	1.0000	0.2500	0.4000
C	0.8066	0.7139	0.7575
C-FAMILY	0.8125	0.4194	0.5532
C-NEGATION	0.8598	0.6484	0.7393
C-SUSPICION	0.5833	0.1014	0.1728
Total	0.8373	0.6942	0.7590
Accuracy	—-	—–	0.9551

Table 4: Evaluation of ntH+CRF all

Tag	Precision	Recall	FB1
A	0.9091	0.7273	0.8081
H	1.0000	0.8493	0.9185
L	0	0	0
T	0.8922	0.8029	0.8452
X	1.0000	0.2500	0.4000
C	0.8053	0.6981	0.7479
C-FAMILY	0.7222	0.4194	0.5306
C-NEGATION	0.8678	0.6402	0.7368
C-SUSPICION	0.6000	0.0870	0.1519
Total	0.8383	0.6834	0.7530
Accuracy	—-	—–	0.9528

Table 5: Evaluation of ntH+CRF-up all

Tag	Precision	Recall	FB1
A	0.7719	0.8000	0.7857
H	0.9118	0.8493	0.8794
L	0	0	0
T	0.8353	0.8353	0.8353
X	1.0000	0.5000	0.6667
C	0.6634	0.6371	0.6500
C-FAMILY	0.4000	0.2581	0.3137
C-NEGATION	0.5711	0.5549	0.5629
C-SUSPICION	0.1905	0.1159	0.1441
Total	66.92	63.79	0.6532
Accuracy	—-	——	0.9361

5.4 Test experiment

We submited 3 models for test evaluation;

- ntH+CRF simple
- ntH+CRF all
- ntH+CRF-up all

The results of test evalution is shown in Table 6 and 7.

Table 6: Test Evaluation Result of de-identification task

Model	Precision	Recall	F-measure	Accuracy
ntH+CRF simple	0.7581	0.7546	0.7564	0.9927
ntH+CRF all	0.7488	0.7454	0.7471	0.9926
ntH+CRF-up all	0.7617	0.7546	0.7581	0.9928

Table 7: Test Evaluation Result of complaint and diagnosis task

Model	Precision	Recall	F-measure	Accuracy
ntH+CRF simple	0.6632	0.6288	0.6456	0.9372
ntH+CRF all	0.6332	0.6071	0.6199	0.9329
ntH+CRF-up all	0.6486	0.6093	0.6283	0.9341

In contrast to the preliminary experiments, the results of ntH+CRF all was clearly bad than ntH+CRF simple. And updating was effective in test evaluation which was not in sample data. These two observations suggests that the results of preliminary experiments were overfitting and updating supervised data to keep consistency might be effective to generalize the model.

6. CONCLUSIONS

We built a general system to recognize special terms from small tagged corpus by integrating bayesian language models and CRF techniques. Especially we improved the recall of de-identification task and suggested the easy method to input generative model outputs to discriminative models.

The scores of our team were not very good, but sometimes recall is more important than precision or F-measure, especially in de-identification application, so we belive ntH is useful in some cases.

7. REFERENCES

[1] Conll-2000 shared task. `http://www.clips.ua.ac.be/conll2000/chunking/`.

[2] Mecab – yet another part-of-speech and morphological analyzer. `http://mecab.sourceforge.jp/`.

[3] D. Mochihashi and E. Sumita. The infinite markov model. *NIPS*, 2007:1–2, 2007.

[4] D. Mochihashi, T. Yamada, and N. Ueda. Bayesian unsupervised word segmentation with hierachical language modeling. *ACL*, 1(36):49, 2009.

[5] M. Morita, Y. Kano, T. Ohkuma, M. Miyabe, and E. Aramaki. Overview of the ntcir-10 mednlp task. In *Proceedings of NTCIR-10*, 2013.

[6] Y. W. Teh. A hierarchical bayesian language model based on pitman-yor processes. *ACL*, pages 985–992, 2006.