A Machine Learning Approach to Acronym Generation

Yoshimasa Tsuruoka[†‡]
[†]CREST
Japan Science and Technology Agency
Japan

Sophia Ananiadou
School of Computing
Salford University
United Kingdom

Jun'ichi Tsujii[‡†]
[‡]Department of Computer Science
The University of Tokyo
Japan

tsuruoka@is.s.u-tokyo.ac.jp
S.Ananiadou@salford.ac.uk
tsujii@is.s.u-tokyo.ac.jp

Abstract

This paper presents a machine learning approach to acronym generation. We formalize the generation process as a sequence labeling problem on the letters in the definition (expanded form) so that a variety of Markov modeling approaches can be applied to this task. To construct the data for training and testing, we extracted acronym-definition pairs from MEDLINE abstracts and manually annotated each pair with positional information about the letters in the acronym. We have built an MEMM-based tagger using this training data set and evaluated the performance of acronym generation. Experimental results show that our machine learning method gives significantly better performance than that achieved by the standard heuristic rule for acronym generation and enables us to obtain multiple candidate acronyms together with their likelihoods represented in probability values.

1 Introduction

Technical terms and named-entities play important roles in knowledge integration and information retrieval in the biomedical domain. However, spelling variations make it difficult to identify the terms conveying the same concept because they are written in different manners. Acronyms constitute a major part of spelling variations (Nenadic et al., 2002), so proper management of acronyms leads to improved performance of the information systems in this domain.

As for the methods for recognizing acronym-definition pairs from running text, there are many studies reporting high performance (e.g. over 96% accuracy and 82% recall) (Yoshida et al., 2000; Nenadic et al., 2002; Schwartz and Hearst, 2003; Zahariev, 2003; Adar, 2004). However, another aspect that we have to consider for efficient acronym management is to generate acronyms from the given definition (expanded form).

One obvious application of acronym generation is to expand the keywords in information retrieval. As reported in (Wren et al., 2005), for example, you can retrieve only 25% of the documents concerning the concept of "JNK" by using the keyword "c-jun N-terminal kinase". In more than 33% of the documents the concept is written with its acronym "JNK". To alleviate this problem, some research efforts have been devoted to constructing a database containing a large number of acronym-definition pairs from running text of biomedical documents (Adar, 2004).

However, the major problem of this database-building approach is that building the database offering complete coverage is nearly impossible because not all the biomedical documents are publicly available. Although most of the abstracts of biomedical papers are publicly available on MEDLINE, there is still a large number of full-papers which are not available.

In this paper, we propose an alternative approach

Proceedings of the ACL-ISMB Workshop on Linking Biological Literature, Ontologies and Databases: Mining Biological Semantics, pages 25–31, Detroit, June 2005.

to providing acronyms from their definitions so that we can obtain acronyms without consulting acronym-definition databases.

One of the simplest way to generate acronyms from definitions would be to choose the letters at the beginning of each word and capitalize them. However, there are a lot of exceptions in the acronyms appearing in biomedical documents. The followings are some real examples of the definition-acronym pairs that cannot be created with the simple heuristic method.

RNA polymerase (RNAP)
antithrombin (AT)
melanoma cell adhesion molecule (Mel-CAM)
the xenoestrogen 4-tert-octylphenol (t-OP)

In this paper we present a machine learning approach to automatic generation of acronyms in order to capture a variety of mechanisms of acronym generation. We formalize this problem as a sequence labeling task such as part-of-speech tagging, chunking and other natural language tagging tasks so that common Markov modeling approaches can be applied to this task.

2 Acronym Generation as a Sequence Labeling Problem

Given the definition (expanded form), the mechanism of acronym generation can be regarded as the task of selecting the appropriate action on each letter in the definition.

Figure 1 illustrates an example, where the definition is "Duck interferon gamma" and the generated acronym is "DuIFN-gamma". The generation proceeds as follows:

> The acronym generator outputs the first two letters unchanged and skips the following three letters. Then the generator capitalizes 'i' and skip the following four letters...

By assuming that an acronym is made up of alphanumeric letters, spaces and hyphens, the actions being taken by the generator are classified into the following five classes.

- SKIP

  The generator skips the letter.

- UPPER

  If the target letter is uppercase, the generator outputs the same letter. If the target letter is lowercase, the generator coverts the letter into the corresponding upper letter.

- LOWER

  If the target letter is lowercase, the generator outputs the same letter. If the target letter is uppercase, the generator coverts the letter into the corresponding lowercase letter.

- SPACE

  The generator convert the letter into a space.

- HYPHEN

  The generator convert the letter into a hyphen.

From the probabilistic modeling point of view, this task is to find the sequence of actions $t_1...t_n$ that maximizes the following probability given the observation $o = o_1...o_n$

$$P(t_1...t_n|o). \tag{1}$$

Observations are the letters in the definition and various types of features derived from them. We decompose the probability in a left-to-right manner.

$$P(t_1...t_n|o) = \prod_{i=1}^{n} p(t_i|t_1...t_{i-1}o). \tag{2}$$

By making a first-order markov assumption, the equation becomes

$$P(t_1...t_n|o) = \prod_{i=1}^{n} p(t_i|t_{i-1}o). \tag{3}$$

If we have the training data containing a large number of definition-acronym pairs where the definition is annotated with the labels for actions, we can estimate the parameters of this probabilistic model and the best action sequence can be efficiently computed by using a Viterbi decoding algorithm.

In this paper we adopt a maximum entropy model (Berger et al., 1996) to estimate the local probabilities $p(t_i|t_{i-1}o)$ since it can incorporate diverse types of features with reasonable computational cost. This modeling, as a whole, is called Maximum Entropy Markov Modeling (MEMM).

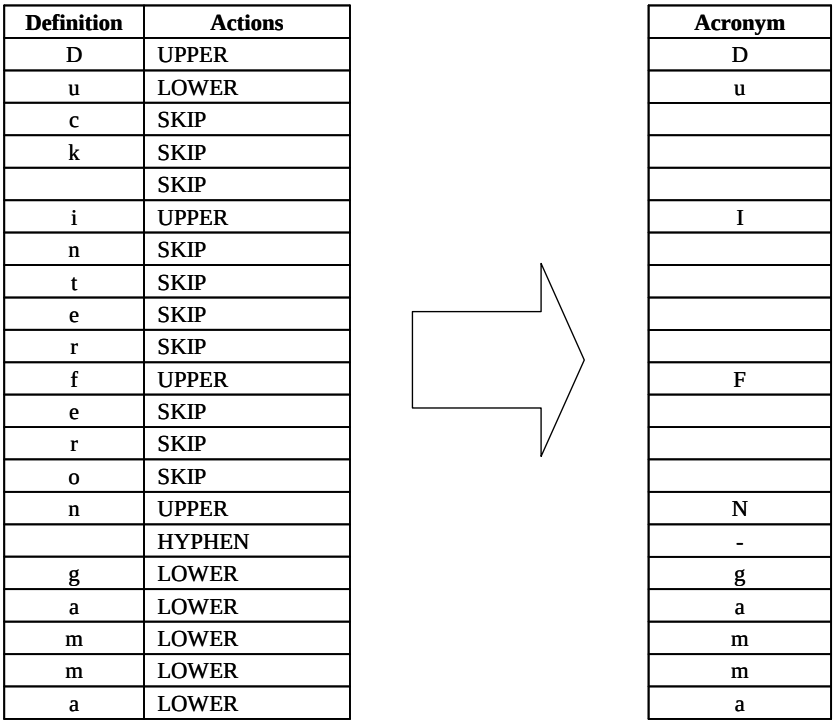

Figure 1: Acronym generation as a sequence labeling problem. The definition is "Duck interferon gamma" and the acronym is "DuIFN-gamma". Each letter in the acronym is generated from a letter in the definition following the action for the letter.

Regularization is important in maximum entropy modeling to avoid overfitting to the training data. For this purpose, we use the maximum entropy modeling with inequality constraints (Kazama and Tsujii, 2003). The model gives equally good performance as the maximum entropy modeling with Gaussian priors (Chen and Rosenfeld, 1999), and the size of the resulting model is much smaller than that of Gaussian priors because most of the parameters become zero. This characteristic enables us to easily handle the model data and carry out quick decoding, which is convenient when we repetitively perform experiments. This modeling has one parameter to tune, which is called *width factor*. We set this parameter to be 1.0 throughout the experiments.

3 The Data for Training and Testing

Since there is no training data available for the machine learning task described in the previous section, we manually created the data. First, we extracted definition-acronym pairs from MEDLINE abstracts using the acronym acquisition method proposed by (Schwartz and Hearst, 2003). The abstracts used for constructing the data were randomly selected from the abstracts published in the year of 2001. Duplicated pairs were removed from the set.

In acquiring the pairs from the documents, we focused only on the pairs that appear in the form of

... *expanded_form* (*acronym*) ...

We then manually removed misrecognized pairs and annotated each pair with positional information. The positional information tells which letter in the definition should correspond to a letter in the acronym. Table 1 lists a portion of the data. For example, the positional information in the first pair indicates that the first letter 'i' in the definition corresponds to 'I' in the acronym, and the 12th letter 'm' corresponds to 'M'.

With this positional information, we can create the training data for the sequence labeling task because there is one-to-one correspondence between the sequence labels and the data with positional information. In other words, we can determine the ap-

Definition	Acronym	Positional Information
intestinal metaplasia	IM	1, 12
lactate dehydrogenase	LDH	1, 9, 11
cytokeratin	CK	1, 5
cytokeratins	CKs	1, 5, 12
Epstein-Barr virus	EBV	1, 9, 14
30-base pairs	bp	4, 9
in-situ hybridization	ISH	1, 4, 9
:	:	:

Table 1: Curated data containing definitions, their acronyms and the positional information.

propriate action for each letter in the definition by comparing the letter with the corresponding letter in the acronym.

4 Features

Maximum entropy modeling allows us to incorporate diverse types of features. In this paper we use the following types of features in local classification. As an example, consider the situation where we are going to determine the action at the letter 'f' in the definition "Duck interferon gamma".

- Letter unigram (UNI)

  The unigrams of the neighboring letters. (i.e. "uni_{-1} r", "uni_0 f", and "uni_{+1} e")

- Letter bigram (BI)

  The bigrams of the neighboring letters. (i.e. "bi_{-2} er", "bi_{-1} rf", "bi_0 fe", and "bi_{+1} er")

- Letter trigram (TRI)

  The trigrams of the neighboring letters. (i.e. "tri_{-2} ter", "tri_{-1} erf", "tri_0 rfe", "tri_{+1} fer", and "tri_{+2} ero")

- Action history (HIS)

  The preceding action (i.e. SKIP)

- Orthographic features (ORT)

  Whether the target letter is uppercase or not (i.e. false)

- Definition Length (LEN)

  The number of the words in the definition (i.e. "len=3")

- Letter sequence (SEQ)

  1. The sequence of the letters ranging from the beginning of the word to the target letter. (i.e. "seq_{left} interf")
  2. The sequence of the letters ranging from the target letter to the end of the word. (i.e. "seq_{right} feron")
  3. The word containing the target letter. (i.e. "seq_{word} interferon")

- Distance (DIS)

  1. The distance between the target letter and the beginning of the word. (i.e. "dis_{left} 6")
  2. The distance between the target letter and the tail of the word. (i.e. "dis_{right} 5")

Rank	Probability	String
1	**0.779**	**TBI**
2	0.062	TUBI
3	0.028	TB
4	0.019	TbI
5	0.015	TB-I
6	0.009	tBI
7	0.008	TI
8	0.007	TBi
9	0.002	TUB
10	0.002	TUbI
ANSWER		TBI

Table 2: Generated acronyms for "traumatic brain injury".

5 Experiments

To evaluate the performance of the acronym generation method presented in the previous section, we ran five-fold cross validation experiments using the manually curated data set. The data set consists of 1,901 definition-acronym pairs.

For comparison, we also tested the performance of the popular heuristics for acronym generation in which we choose the letters at the beginning of each word in the definition and capitalize them.

Rank	Probability	String
1	0.423	ORF1
2	0.096	OR1
3	**0.085**	**ORF-1**
4	0.070	RF1
5	0.047	OrF1
6	0.036	OF1
7	0.025	ORf1
8	0.019	OR-1
9	0.016	R1
10	0.014	RF-1
ANSWER		ORF-1

Table 3: Generated acronyms for "open reading frame 1".

Rank	Probability	String
1	0.163	RNA-P
2	0.147	RP
3	0.118	RNP
4	**0.110**	**RNAP**
5	0.064	RA-P
6	0.051	R-P
7	0.043	RAP
8	0.041	RN-P
9	0.034	RNA-PM
10	0.030	RPM
ANSWER		RNAP

Table 4: Generated acronyms for "RNA polymerase".

5.1 Generated Acronyms

Tables 2 to 5 show some examples of generated acronyms together with their probabilities. They are sorted with their probabilities and the top ten acronyms are shown. The correct acronym given in the training data is described in the bottom row in each table.

In Table 2, the definition is "traumatic brain injury" and the correct acronym is "TBI". This is the simplest case in acronym generation, where the first letter of each word in the definition is to be capitalized. Our acronym generator gives a high probability to the correct acronym and it is ranked at the top.

Table 3 shows a slightly more complex case, where the generator needs to convert the space be-

Rank	Probability	String
1	0.405	MCPP
2	0.149	MCP
3	0.056	MCP
4	0.031	MPP
5	0.028	McPP
6	0.024	MchPP
7	0.020	MC
8	0.011	MP
9	**0.011**	**mCPP**
10	0.010	MCRPP
ANSWER		mCPP

Table 5: Generated acronyms for "meta-chlorophenylpiperazine".

Rank	Probability	String
1	0.811	TV
2	0.034	TSV
3	0.030	TCV
4	0.021	Tv
5	0.019	TVs
6	0.013	T-V
7	0.008	TOV
8	0.004	TSCV
9	0.002	T-v
10	**0.001**	**TOSV**
ANSWER		TOSV

Table 6: Generated acronyms for "Toscana virus".

Rank	Coverage (%)
1	55.2
2	65.8
3	70.4
4	73.2
5	75.4
6	76.7
7	78.3
8	79.8
9	81.1
10	82.2
BASELINE	47.3

Table 7: Coverage achieved with the Top N Candidates.

tween ‘F’ and ‘1’ into a hyphen. The correct answer is located at the third rank.

The definition in Table 4 is “RNA polymerase” and the correct acronym is “RNAP”, so the generator needs to the first three letters unchanged. The correct answer is located at the fourth rank, and the probability given the correct answer does not have a large gap with the top-ranked acronym.

Table 5 shows a more difficult case, where you need to output the first letter in lowercase and choose appropriate letters from the string having no delimiters (e.g. spaces and hyphens). Our acronym generator outputs the correct acronym at the nine-th rank but the probability given this acronym is very low compared to that given to the top-ranked string.

Table 6 shows a similar case. The probability given to the correct acronym is very low.

5.2 Coverage

Table 7 shows how much percentage of the correct acronyms are covered if we take top N candidates from the outputs of the acronym generator. The bottom line (BASELINE) shows the coverage achieved by generating one acronym using the standard heuristic rule for acronym generation. Note that the coverage achieved with a single candidate (Rank 1) is better that of BASELINE.

If we take top five candidates, we can have a coverage of 75.4%, which is considerably better than that achieved by the heuristic rule. This suggests that the acronym generator could be used to significantly improve the performance of the systems for information retrieval and information integration.

5.3 Features

To evaluate how much individual types of features affect the generation performance, we ran experiments using different feature types. Table 8 shows the results. Overall, the results show that various types of features have been successfully incorporated in the MEMM modeling and individual types of features contribute to improving performance.

The performance achieved with only unigram features is almost the same as that achieved by the heuristic rule. Note that the features on the previous state improve the performance, which suggests that our selection of the states in the Markov modeling is a reasonable choice for this task.

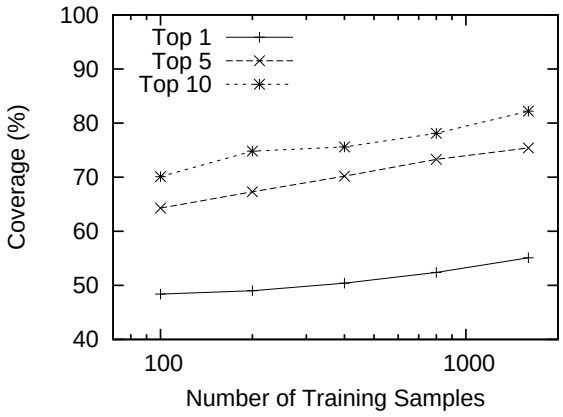

Figure 2: Learning curve.

5.4 Learning Curve

Figure 2 shows a learning curve of our acronym generator, which shows the relationship between the number of the training samples and the performance of the system. The graph clearly indicates that the performance consistently improves as the training data increases and still continues to improve even when the size of the training data reaches the maximum. This suggests that we can achieve improved performance by increasing the annotated data for training.

6 Conclusion

We presented a machine learning approach to acronym generation. In this approach, we regarded the generation process as a sequence labeling problem, and we manually created the data for training and testing.

Experimental results using 1901 definition-acronym pairs, we achieved a coverage of 55.1%, which is significantly bettern than that achieved by the standard heuristic rule for acronym generation. The algorithm also enables us to have other acronym candidates together with the probabilities representing their likelihood.

6.1 Future work

In this paper we did not consider the generation mechanisms where the letters in the acronym appear in a different order in the definition. Since about 3% of acronyms reportedly involve this types of generation mechanism (Schwartz and Hearst, 2003), we

Feature Templates	Top 1 Coverage (%)	Top 5 Coverage (%)	Top 10 Coverage (%)
UNI	48.2	66.2	74.2
UNI, BI	50.1	71.2	78.3
UNI, BI, TRI	50.4	72.3	80.1
UNI, BI, TRI, HIS	50.6	73.6	81.2
UNI, BI, TRI, HIS, ORT	51.0	73.9	80.9
UNI, BI, TRI, HIS, ORT, LEN	53.9	74.6	81.3
UNI, BI, TRI, HIS, ORT, LEN, DIS	54.4	75.0	81.8
UNI, BI, TRI, HIS, ORT, LEN, DIS, SEQ	55.1	75.4	82.2

Table 8: Performance with Different Feature Sets.

might further improve performance by considering such permutation of letters.

As the learning curve (Fig 2) suggested, one obvious way to improve the performance is to increase the training data. The size of the training data used in the experiments is fairly small compared to those in other sequence tagging tasks such POS tagging and chunking. We plan to increase the size of the training data with a semi-automatic way that could reduce the human effort for annotation.

References

Eytan Adar. 2004. Sarad: A simple and robust abbreviation dictionary. *Bioinformatics*, 20(4):527–533.

Adam L. Berger, Stephen A. Della Pietra, and Vincent J. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71.

Stanley F. Chen and Ronald Rosenfeld. 1999. A gaussian prior for smoothing maximum entropy models. *Technical Report CMUCS -99-108, Carnegie Mellon University*.

Jun'ichi Kazama and Jun'ichi Tsujii. 2003. Evaluation and extension of maximum entropy models with inequality constraints. In *Proceedings of EMNLP 2003*.

Goran Nenadic, Irena Spasic, and Sophia Ananiadou. 2002. Automatic acronym acquisition and term variation management within domain-specific texts. In *Proceedings of the LREC-3*, pages 2155–2162.

Ariel Schwartz and Marti Hearst. 2003. A simple algorithm for identifying abbreviation definitions in biomedical texts,. In *Proceedings of the Pacific Symposium on Biocomputing (PSB 2003)*.

Jonathan D. Wren, Jeffrey T. Chang, James Pustejovsky, Eytan Adar, Harold R. Garner, and Russ B. Altman. 2005. Biomedical term mapping databases. *Nucleic Acid Research*, 33.

M. Yoshida, K. Fukuda, and T. Takagi. 2000. Pnad-css: a workbench for constructing a protein name abbreviation dictionary. *Bioinformatics*, 16(2):169–175.

Manuel Zahariev. 2003. An efficient methodology for acronym-expansion matching. In *Proceedings of the International Conference on Information and Knowledge Engineering (IKE)*.