Journal of Bionanoscience
Vol. 7, 66–71, 2013

RESEARCH ARTICLE

An Hidden Markov Model-Based Transcription Factor Mining Method

Chunguo Wu[1, 2, 3], Gaoyang Li[1], Xiaozhou Wu[1], Xiaosong Han[1], Binghong Wang[2], Jesualdo Tomás Fernández Breis[4], Fei Wang[3], and Yanchun Liang[1, *]

[1] *Key Laboratory for Symbol Computation and Knowledge Engineering of National Education Ministry, College of Computer Science and Technology, Jilin University, Changchun 130012, China*
[2] *Business School, University of Shanghai for Science and Technology, Shanghai 200093, China*
[3] *Shanghai Key Laboratory of Intelligent Information Processing, Shanghai 200433, China*
[4] *Department of Informatics and Systems, University of Murcia, 30100, Murcia, Spain*

A text mining algorithm named HMM-TFM (Hidden Markov Model based transcription factor name mining) is presented. The proposed algorithm does not need a dictionary of transcription factor names. A small verb set is defined to filter sentences. Transcription factor names are mined according to the part of speech tagged by hidden Markov model. Experimental results show that the recall and precision of HMM-TFM achieve 74.2% and 77.9%, respectively.

Keywords: Hidden Markov Models, Transcription Factor, Text Mining, Promoter, Bioinformatics.

1. INTRODUCTION

Promoter is a regulatory region of DNA located 5′ upstream of a gene, providing a control point for regulated gene transcription. There are a few functional elements and one transcription start site in a promoter region. Transcription factors (TFs) recognize and bind to the cis-regulatory elements in a promoter region, which is of importance for the implementation of many biological functions in all life entities. Hence, mining TF names from scientific literatures is significant for biological investigation. There is not enough published work on mining/extraction TF names from scientific literatures, although quite a few works have been put on similar biological element discovery. For example, machine learning-based and rule-based text mining approaches are widely used for extracting information of genes/proteins in bioinformatics. Fundel et al. (2005) used synonym lists that map the unique database identifiers for each gene/protein to different synonym names.[1] Hanisch et al. (2005) proposed the ProMiner system using a pre-processed synonym dictionary to identify potential name occurrences in the biomedical text and associate protein and gene database identifiers with the detected matches.[2] Aerts et al. (2008) used a vector space model to identify Medline abstracts from papers, and calculated the cosine similarity measure between individual abstract vector and the composite query vector.[3] Zhou et al. (2007) applied semi-supervised learning of the hidden vector state model for protein-protein interactions extraction.[4] Yang et al. (2006) combined dictionary-based approach and machine learning-based approach, and used edit distance to approximate string searching algorithm to improve the recall rate of gene recognition.[5] Yang et al. (2008) presented a text mining algorithm for TF binding site information extraction based on "question answering" system.[6] Synonym dictionary based approaches need a list of genes/proteins. These approaches are used effectively to detect genes/proteins from scientific literatures, since researchers have accumulated quite a few synonyms of genes and proteins. Unfortunately, there is very little work about meta-item mining in medical or life science fields, especially, about molecular biology. Compared with genes or proteins, there is little information of TF to build a comprehensive synonym repository. Liu et al. (2010) proposed a second-order hidden Markov model and applied it to Chinese part-of-speech tagging problem. This inspires us using HMM to tag the part of speech (POS).[7] The TF names are mined according to the POS.

In this paper, we propose a novel text mining method to extract TF names from English scientific literatures, named HMM-TFM (Hidden Markov Model based TF mining), which is used for extracting words denoted TFs in English scientific literatures.[8–10] Being different from traditional synonym dictionary based approaches, the proposed HMM-TFM does not build any entity dictionary of TFs. In each sentence, HMM-TFM tags the POS for each

*Author to whom correspondence should be addressed.

 1557-7910/2013/7/066/006 doi:10.1166/jbns.2013.1090

word based on the hidden Markov model (HMM). In the proposed method, both the hidden state and the observation are the POS, but they are obtained differently. The observation sequences are obtained by the suffix of each word in sentences. The hidden states are the real POS for words, which are achieved from decoding. We construct a set called *predicate* consisting of eight appointed verbs, which are usually used as predicate connecting a TF and a promoter. This predicate set is used for filtering sentences. The filtered sentences are most probable to contain TFs. And then, we score each word according to both its POS and its relationship with the verbs in the *predicate* set. Words with aggregate score higher than a predetermined threshold θ are identified as TFs.

2. HMM-BASED TF MINING ALGORITHM

Hidden Markov model (HMM) is specified by five elements $(\Omega_X, \Omega_O, A, B, \pi)$, where $\Omega_X = \{X_1, \ldots, X_N\}$ is the set of hidden states, and $\Omega_O = \{O_1, \ldots, O_M\}$ is the set of observations. $A = \{a_{ij}\}$ is the state transition probability matrix, $a_{ij} = p(q_{t+1} = x_j \mid q_t = x_i)$ represents the probability of moving from the state i to the state j. $B = \{b_i(k)\}$ is the observation likelihood matrix, $b_i(k) = p(o_t = v_k \mid q_t = x_i)$ expresses the probability of an observation v_k being generated from a state x_i. $\pi = \{\pi_1, \ldots, \pi_N\}$ is the initial state probability vector, $\pi_i = p(q_1 = x_i)$ represents the probability of initial state being x_i.[8–12] For convenience, HMM is usually represented as three elements $\lambda = \{A, B, \pi\}$.

2.1. Observation Set of HMM-TFM

The first step of HMM-TFM is to define the observation set. For some high-frequency words, including conjunctions, prepositions and copulas in natural language, the observations are their real POS. For the other words, their POS is presumed according to the suffix, since the suffix could indicate the POS more reliably in linguistics sense. In this paper, the observation set is as follows:

$$\Omega_O = \{verb, adv, aux, conj, punctuation, art, prep, adj, be, num, pron, noun, unknown\}$$

where $verb$ represents the observation of the verb, adv the adverb, aux the auxiliary word, $conj$ the conjunction, art the article, $prep$ the preposition, adj the adjective, be the copula, num the numeral, $pron$ the pronoun, $noun$ the noun. Ignoring the punctuation will make grammatical structure of sentences difficult to distinguish, thereby $punctuation$ is considered as an observation. For the words those are not used frequently in natural language, and could not be presumed the POS according to their suffix, for example, the word "blot," the observation is $unknown$.

The relationship of mapping high-frequency words and suffixes to observations is shown in Table I. By this table, we can confirm the POS of most words. For example, words with the suffix "-ment" are mapped to the observation *noun*, words with suffix "-ate" are mapped to the observation $verb$. There are totally 1095 words in training samples and only 370 words are mapped to the observation $unknown$. In this way, each sentence in literatures is treated as an observation sequence.

Table I. The high-frequency word and suffix of observations.

Observation	High-frequency word	Suffix
verb	do, did	-ate, -fy, -ish, -ize, -ise, -ine, -en, -ed
adv		-ly, -ward, -ways, -wise, -style, -fold
aux	can, should, would, will, shall, could, may, must	
conj	whether, that, but, which, and, or, while, so, because, since	
art	a, an, the	
prep	of, on, at, in, after, with, through, for, to, by, from, about	
adj		-ous, -ant, -ent, -al, -able, -ible, -ar, -ic, -ical, -ary, -an, -ary, -ern, -ese, -ful, -ist, -ive, -less, -like
be	be, been, am, is, are, was, were	
pron	it, he, she, they, I, we, you, us, them, this, these	
noun		-ion, -or, -er, -ment, -ence, -ship, -ness, -ism, -ity, -pathy, -osis, -oma, -emia, -megaly, -ectomy, -ol

It is obvious that it is impossible to estimate accurately the POS only by using suffixes. There are many words with multiple POS. In this paper, HMM-TFM selects the most frequently-used POS as the observation for each word. In fact, the POS is identified finally according to the hidden state, but not the observation. The hidden state is calculated from both the observations and the adjacent hidden states in decoding. For example, the word "implement" can be used as noun or verb. HMM-TFM maps this word to the observation $noun$. In the observation likelihood matrix B, the probability of the observation $noun$ being generated from the hidden state $verb$ is not zero. Hence, it is possible that the hidden state of the word "implement" transfers from the state $noun$ to the state $verb$.

2.2. Hidden State Set of HMM-TFM

The hidden state is defined as the real POS. The set of hidden states is in the form as follows:

$$\Omega_X = \{verb, adv, aux, conj, punctuation, art, prep, adj, be, num, pron, noun, tf\}$$

RESEARCH ARTICLE

where the state tf represents the TF, and the other states are the same as those in the observation set, but gotten from different methods. The observation presumed according to the suffix. In the learning process, the hidden states are marked artificially, and in the decoding process, they are identified according to the observation sequences and the specific HMM $\lambda = \{A, B, \pi\}$.

Given an observation sequence O and an HMM $\lambda = \{A, B, \pi\}$, finding the best hidden state sequence Q is a kind of dynamic programming. The best hidden state sequence Q makes $P(Q \mid O, \lambda)$ maximal. We induct two measures for obtaining the best hidden state sequence Q. $\delta_t(i)$ represents the probability that the HMM is in state i after seeing the first t observations and passing through the most likely state sequence $q_1, ..., q_{t-1}$. $\psi_t(i)$ represents the state which state i moves from. Like other dynamic programming algorithms, $\delta_t(i)$ can be calculated recursively. Given the probability of being in each state at time $t-1$, we compute the $\delta_t(i)$ by taking the most probable of the path extensions leading to the current state.

We can give a formal process for calculating the best hidden state sequence Q:

(1) Initialization:

$$\delta_1(i) = \pi_i b_i(o_1), \quad \psi_1(i) = 0 \tag{1}$$

(2) Recursion:

$$\begin{aligned} \delta_t(i) &= \max_j \delta_{t-1}(j) a_{ji} b_i(o_t) \\ \psi_t(i) &= \arg\max_j [\delta_{t-1}(j) a_{ji}] \end{aligned} \tag{2}$$

(3) Termination:

$$P(Q \mid O, \lambda) = \max_i [\delta_T(i)] \tag{3}$$

(4) Backtrace:

$$q_t = \psi_{t+1}(q_{t+1}) \tag{4}$$

In this way we can obtain the hidden state of each word in sentences. We mine TF names by scoring each word based on the hidden state. The score of TFs is supposed to be the highest among all words. In order to give prominence to the TFs, the score of the state tf is 7, the score of the state *noun* is 5, and the score of the rest states is only 1. HMM-TFM sums the aggregate score of each specific word sentence by sentence. The word with an aggregate score greater than the predetermined threshold θ is identified as a TF.

2.3. States Combining and Sequences Filtering

In the process of decoding hidden states from observations, the time complexity of calculating $P(Q \mid O, \lambda)$ is $\Theta(TN^2)$, and the space complexity is $\Theta(TN)$, where N is the size of the hidden state set. Obviously, the smaller N is, the more efficient the HMM-TFM could be implemented. The size of both observation set and hidden state set is eleven for the proposed HMM-TFM. Combination of elements in hidden state set can reduce the size N. In addition, if the size of the observation set and hidden state set is not more than ten, we can use number 0∼9 to represent the hidden states and the observations in a given sequence, which will bring convenience in dealing with both the hidden state sequences and the observation sequences. Hence, we attempt to combine elements in the hidden state set and the observation set.

Given an HMM $\lambda = \{A, B, \pi\}$, a couple of hidden states, X_i and X_j, with following characters can be combined into a new state X'_j:

$\forall o_1 o_2 \ldots o_t, \exists k,$

$$\begin{aligned} &\forall l, P\{o_1 \ldots o_t, q_t = X_l, o_t = O_k \mid \lambda\} \\ &\quad < P\{o_1 \ldots o_t, q_t = X_i, o_t = O_k \mid \lambda\} \end{aligned} \tag{5}$$

$$\begin{aligned} &\exists j, P\{o_1 \ldots o_t, q_t = X_j, o_t \neq O_k \mid \lambda\} \\ &\quad > P\{o_1 \ldots o_t, q_t = X_i, o_t \neq O_k \mid \lambda\} \end{aligned} \tag{6}$$

And then the HMM parameters need to be adjusted as follows:

$$\pi_{j'} = \pi_i + \pi_j \tag{7}$$

$$a_{sj'} = a_{si} + a_{sj}, \quad a_{j's} = a_{is} + a_{js} \tag{8}$$

$$b_{j's} = b_{is} + b_{js} \tag{9}$$

where $\forall s \in \{1, 2, \ldots, M\}$, $s \neq i, j$.

Similar to the state combination, a couple of observations, O_k and O_h, with the following character can be combined into a new observation O'_h:

$\forall o_1 o_2 \ldots o_t, \ \exists h, \ h \neq k,$

$$\begin{aligned} &\forall l, P\{o_1 \ldots o_t, q_t = X_l, o_t = O_h \mid \lambda\} \\ &\quad < P\{o_1 \ldots o_t, q_t = X_j, o_t = O_h \mid \lambda\} \end{aligned} \tag{10}$$

And then the observation likelihood matrix B needs to be adjusted as follows:

$$b_{sh'} = b_{sk} + b_{sh} \tag{11}$$

where $\forall s \in \{1, 2, \ldots, M\}$ and $s \neq h, k$.

In the matrix B, $b_{aux}(aux) \approx 1$ meets the inequality (5). In the matrix A, for any state, $\forall l, a_{l,\,adv} > a_{l,\,aux} (X_l \neq adv, aux)$ meets the inequality (6). So HMM-TFM combines the state adv and the state aux. What's more, in the matrix B, $b_{adv}(adv) \approx 1$ meets the inequality (12). So the observations adv and aux can be combined. In the same way, num and adj, $pron$ and $noun$ can be combined.

After combination, the hidden state set and the observation set are as follows:

$$\begin{aligned} \Omega_X = \{&verb,\ adv \text{ and } aux,\ conj,\ punctuation \\ &art,\ prep,\ adj,\ be, noun,\ tf\} \\ \Omega_O = \{&verb,\ adv \text{ and } aux,\ conj,\ punctuation \\ &art,\ prep,\ adj,\ be,\ noun,\ unknown\} \end{aligned}$$

In this paper, we are interested in the sentences describing the relationship between a TF and a promoter. HMM-TFM uses a strategy based on a verb set called *predicate* to filter sentences related to TFs and promoters. In this verb set, there are 8 verbs, all of which can be used for expressing the relationship between TFs and promoters. There must be a element of the *predicate* set in the filtered sentences. It is most possible that a TF appears as the subject and a region of promoter appears as the object of the filtered sentences. The verb set *predicate* is collected artificially from 50 literatures related to TFs. Compared with the entity dictionary of TFs, the verb set is very small. Hence, it is possible to collect the verb set manually. The verb set *predicate* is as follows:

$$predicate\{repress, bind, transactivate, regulate, activate, suppress, upregulate, downregulate\}$$

In a sentence, the object is either at the position behind the predicate (for passive voice, object is in front of predicate), or in the object clause. In order to reducing the length of the observation sequences, the words behind the appointed predicate have the negative score. In this way, it is not necessary to calculate the hidden state of the words behind the predicate in verb set (for passive voice, in front of the predicate). This strategy reduces the length of the observation sequences, thereby reducing the time complexity and space complexity of the HMM-TFM.

HMM-TFM mines TFs via recognizing the POS of each word. It is necessary to sustain a small size verb set, two matrixes of the state transition probability and the observation likelihood, and an initial state probability vector. The cost is much lower than using of entity dictionary of TFs.

3. EXPERIMENTAL DETAILS

We use TF and promoter as the key words to search literatures manually from a database of scientific literatures. Totally, we retrieve 50 scientific literatures to construct the training set. After being performed the sentence filtering, 969 sentences are extracted from these literatures, which contain the appointed verbs as predicates. These sentences are tagged manually the POS of each word. Thereby the hidden state sequences are generated.[11, 12] The observation sequences are generated as described in Section 2.1. The probability of initial state is

$$\pi = \{0.07018, 0.16236, 0.09297, 0.00117, 0.16207, 0.11643, 0.06965, 0.00029, 0.23178, 0.09309\}$$

The state transition probability matrix and the observation likelihood matrix of the trained HMM $\lambda = \{A, B, \pi\}$ are shown in Tables II and III.

Since the lengths of literatures are not uniform, it is not advisable to choose a fixed threshold. For each literature, we calculate 80% of the maximum value among all words as threshold θ. The results show that the score of TF is much great than the score of other words. The score difference of two TFs described in the same literature is small.

We use transcription factor as key word to choose another 150 literatures from PubMed to test the performance of HMM-TFM and tag 190 TFs manually from these literatures. HMM-TFM mines 181 words as TFs, 141 of which are the same as those tagged manually. The experimental results show that the recall and the precision rates of HMM-TFM achieve 74.2% and 77.9%, respectively. The manually selected and automatically identified TFs are shown in Table IV.

We use F-measure to compare HMM-TFM with similar methods. F-measure is computed according to the recall and precision.

$$F\text{-}measure = \frac{2 \times precision \times recall}{precision + recall} \tag{12}$$

Fundel et al. (2005) used synonym lists to identify gene/protein names.[1] The F-measures are 0.764 for mouse and 0.768 for fly. Yang et al. (2008) tagged the POS artificially and extracted the information of the transcription factor binding site based on the QA system.[6] The F-measure is 0.753. In this article, the F-measure of HMM-TFM is 0.76. The experimental results show that the cost of HMM-TFM is much less than similar methods, but the performance is as well as the similar methods.

Table II. State transition probability matrix.

	verb	*adv* and *aux*	*conj*	*punc*	*art*	*prep*	*adj*	*be*	*noun*	*tf*
verb	0.025	0.111	0.4899	0.0499	0.0621	0.1843	0.0254	0.001	0.0377	0.0132
adv and *aux*	0.5296	0.0875	0.0394	0.2028	0.0202	0.0106	0.0298	0.0491	0.0106	0.0202
conj	0.111	0.0743	0.001	0.001	0.0499	0.0132	0.0621	0.001	0.0865	0.5999
punc	0.121	0.016	0.046	0.046	0.091	0.031	0.106	0.001	0.346	0.196
art	0.001	0.0398	0.001	0.001	0.001	0.001	0.3504	0.001	0.6027	0.001
prep	0.1641	0.0359	0.0359	0.001	0.2223	0.001	0.1175	0.001	0.2106	0.2106
adj	0.001	0.001	0.0467	0.0314	0.001	0.0619	0.0162	0.001	0.7625	0.0772
be	0.221	0.331	0.001	0.056	0.056	0.001	0.166	0.001	0.111	0.056
noun	0.2279	0.0423	0.0783	0.1196	0.0062	0.2176	0.0371	0.0474	0.1815	0.0423
tf	0.2436	0.3795	0.0786	0.1175	0.001	0.0592	0.0204	0.0398	0.0593	0.001

Table III. Observation likelihood matrix.

	verb	*adv* and *aux*	*conj*	*punc*	*art*	*prep*	*adj*	*be*	*noun*	*unknown*
verb	0.7798	0.0133	0.0009	0.00117	0.0009	0.0013	0.0066	0.00029	0.00155	0.1941
adv and *aux*	0.0496	0.768	0.00088	0.00117	0.0091	0.0013	0.00058	0.00029	0.0792	0.0981
conj	0.00111	0.0012	0.6568	0.00117	0.0091	0.0136	0.00058	0.0003	0.01392	0.3104
punc	0.00111	0.0012	0.00088	0.9911	0.0009	0.0013	0.0006	0.0003	0.00155	0.001
art	0.0011	0.0012	0.0009	0.00012	0.991	0.00132	0.00058	0.0003	0.00155	0.00099
prep	0.0244	0.001199	0.001	0.0012	0.0009	0.8865	0.0006	0.00029	0.0248	0.0592
adj	0.2296	0.001199	0.00088	0.00117	0.00091	0.00132	0.397	0.00029	0.032	0.3361
be	0.0561	0.0012	0.00088	0.00117	0.00091	0.0013	0.00058	0.9353	0.0015	0.001
noun	0.0115	0.0012	0.00088	0.00117	0.0009	0.00132	0.0318	0.0003	0.403	0.5481
tf	0.0011	0.001	0.0009	0.00117	0.0009	0.00132	0.00058	0.00029	0.0016	0.99

Table IV. TFs in the 150 literatures.

PMID	TFs		PMID	TFs		PMID	TFs	
10982849	**KKLF**	**MAZ**	21600882	**cAMP**	Sp1	21734707	**Nrf2**	
21498677	**WRKY33**		21642992	NF-YA		21565167	**c-Jun**	ATF2
21208406	**KLF15**		21506108	**FGF-2**		21277289	**Nrf2**	
20840862	Foxo	HSF	21574244	**IRF6**		21277915	**Sigma**	
20473858	**DNMT3a**		21470566	Sp1		21703547	**c-Jun**	**AP-1**
21408062	HSF-1		21401746	**CrMYC2**		21497567	STAT3	
21294160	**Dox**		21623364	IEGs		21329726	**SRF**	
21225257	**E2F1**		21698120	**PITX3**		21731748	**MeCP2**	**MYCN**
21329726	**SRF**		21695256	**CREB**		21731707	**Sp1**	NF-kB
21069812	**FGF-2**		21107995	Sigma		21399967	**Nrf2**	
21536231	**Sry**		21506129	**DEC1**		21738690	**MTF-1**	
21625432	**E2F1**		21399967	**Nrf2**	Oct1	21731766	STAT3	
21653923	EWS-FLI1		21362508	**p150**		22046507	**NOX**	**ROS**
21362474	**Elk-1**		21392589	**CRX**		21433063	Rel	
21455770	**HSF1**		21362510	**Sp1**	**Sp3**	21573184	**Src**	**EGFR**
21136273	**BMP-6**		21342666	JHBP		22046437	**CARM1**	
21637323	**SRY**		21695171	**CXCL12**	**CXCR4**	21738685	HIF-1α	
21603612	**Sp1**		21515911	**NF-kB**	AP-1	21530485	**AP-1**	
21467583	**ABI3**		21632490	**CnAβ1**		21558273	C/EBPβ	
21487097	**SPL9**		21549717	**NF-Y**		21672091	**Sp3**	CADM1
21374086	RD29A	RD29B	21507677	**LPS**		17195167	**Nanog**	**Oct4**
21504520	C/EBP		21565167	**c-Jun**	ATF2	21555002	**EVI1**	
21241485	**Brn-3b**		21399967	**Nrf2**	Oct1	21455926	Tbx3	Runx2
21530491	**PARP-1**		21317191	**Rev-Erbα**	**RORγ**	21705428	**p300**	
21252119	**ATRA**	**PDGF-BB**	21310710	**EBPβ**	**SATB1**	21705419	**factor-YA**	
21655213	Glass		21486745	TFs		21497119	**CREB1**	MAPK
21637838	**Notch**		21542860	**ToxR**		21530336	**KLF2**	
21625455	**TGFβ1**		21542864	**CRP**	**Fis**	21698120	**PITX3**	
21602176	**NFI-A**		21315474	IDEF1		21655251	Hsp	
21279417	**FXR**		21666495	**BDNF**	**NF-kB**	21510935	**FoxO**	
21288884	**Hmo1**		21554857	**Sp1**	**Sp3**	21527253	**RANK**	STAT1
21278163	TEFM		22045596	Gli2b		20709022	**KLF15**	
21307385	**E2F**		21573214	**HEN2**	**HES1**	22037307	**CycE**	
21467577	**API5**		21381079	**EGFR**		21447339	C/EBP	
20875471	**HDACs**		21152987	**CREB**		11473619	**MAZ**	
21506120	**SRF**		21569840	Nrf2		22038624	**CaSR**	
21396685	FVIII	FIX	21491543	Dox		22039015	**Gata3**	
21541973	**NF-κB**	**c-myc**	21495113	**Sox6**		21677782	**NKX2-1**	HOXB5
21394759	**Mecp2**		21518253	**HMGN2**		22039435	**Myc**	
21466784	**Foxa2**		21258403	**EAPP**		22039477	**Pitx2c**	
21258408	**SRF**		21399658	**ISRE**	Ezh2	22046564	**Ikaros**	
20868272	**TFs**		21440621	CD34		21336627	**CRM**	TRF2
21241068	**Snail**		21399922	**IRF-3**		22046555	**MITF**	
21484256	**HCF-1**		21561061	**AP-1**		22046413	**hPaf1**	**PD2**
21547450	DRNL		21406062	**IGF-1**	**YY1**	22046379	**ATF3**	
15277472	**KLF15**		22046091	CRF		22046352	**Sox9**	
21459369	**IL-8**		21674059	**MDM2**	**RITA**	21966511	**PPARβ/δ**	ANGPTL4
21439021	**c-Fos**		21673954	**NS3**	SRCAP	22046282	**Gadd153**	
21470923	IL-6	STAT3 **JAK2**	21550660	**p21**		22018489	**HSF1**	
21457294	**FlbD**		21600798	**Eomes**		22017871	**Ofd1**	

Note: The bold ones are those identified by the HMM-TFM.

4. CONCLUSIONS AND DISCUSSIONS

In this paper, we propose an HMM-based TF mining algorithm, named HMM-TFM, which maps each word to an observation by suffix and then, accordingly, decodes the hidden state. HMM-TFM scores each word according to its hidden state. The words with aggregate score greater than 80 percent of the maximum aggregate score are identified as TFs. The innovation of the proposed HMM-TFM is the strategy of filtering sentences by a set of appointed verb. On one hand, this strategy makes use of the words in a sentence to presume the POS, instead of computing the edit distance word by word. On the other hand, the sequences with no contribution to the anticipative target are removed at the earlier stage, which improve further the efficiency of the proposed method. The experimental testing results of 150 literatures randomly selected from PubMed show that the recall and precision of the proposed HMM-TFM achieve 74.2% and 77.9% respectively. The future work is to make the verb set expanded automatically by using the identified TFs and promoters, which would enable the proposed method to run circularly.

Acknowledgment: The authors are grateful to the support of NSFC (grant 60703025, 91024026), "985" and "211" project, and Shanghai Key Laboratory of Intelligent Information Processing open project (IIPL-2011-006).

References and Notes

1. K. Fundel, D. Guttler, R. Zimmer, and J. Apostolakis, *BMC Bioinformatics* 6, S14 **(2005)**.
2. D. Hanisch, K. Fundel, H. T. Mevissen, R. Zimmer, and J. Fluck, *BMC Bioinformatics* 6, S14 **(2005)**.
3. S. Aerts, M. Haeussler, S. Vooren, O. L. Griffith, P. Hulpiau, S. J. Jones, S. B. Montgomery, and C. M. Bergman, *Gen. Biol.* 9, R31.1 **(2008)**.
4. D. Zhou, Y. He, and C. K. Kwoh, *Artif. Intell. Med.* 4, 64 **(2007)**.
5. Z. H. Yang, H. F. Lin, and J. Zhao, *Proc. Sixth World Congress on Intelligent Control and Automation*, Dalian, China, June **(2006)**, p. 9391.
6. Q. Yang, G. Y. Zheng, Y. Xiong, and Y. Zhu, *J. Comput. Res. Dev.* 45, 323 **(2008)**.
7. J. B. Liu, M. Q. Song, F. Zhao, and Z. Y. Yang, *Eng. Comput.* 36, 231 **(2010)**.
8. J. Goutsias, *IEEE ACM T. Comput. Bi.* 3, 57 **(2006)**.
9. Q. Liu, Y. S. Zhu, and B. H. Wang, *Comput. Biol. Chem.* 27, 69 **(2003)**.
10. R. Y. Kahsay, G. Gao, and L. Liao, *Bioinformatics*, 21, 1853 **(2005)**.
11. L. Y. Zou, *Comput. Biol. Med.* 40, 621 **(2010)**.
12. J. R. Otterpohl, *Lecture Notes in Computer Science* 2415, 1180 **(2002)**.

Received: 28 March 2012. Accepted: 2 June 2012.

RESEARCH ARTICLE