2010 22nd International Conference on Tools with Artificial Intelligence

Language Model Combination for Community-based Q&A Retrieval

Akira Takahashi
University of Tokyo
Tokyo 101–8430, Japan
Email: a-takahashi@10.alumni.u-tokyo.ac.jp

Atsuhiro Takasu and Jun Adachi
National Institute of Informatics
Tokyo 101–8430, Japan
Email: {takasu,adachi}@nii.ac.jp

***Abstract*—This paper proposes three methods for combining various probabilistic models for retrieving answers from community-based question answering (cQA) archives. We adopt four probabilistic models for these combinations, i.e., (1) the language model measuring similarity between a query and a question stored in the cQA archive, (2) two translation models for measuring the similarity between a query and an answer stored in the cQA archive, and a background language model for smoothing. Then, we developed three parameter estimation methods. Two of them are mixture models of the language models. The remaining model exploits the difference between the models. We apply the proposed methods to a cQA archive and show that they significantly outperform a widely used language model and Okapi BM25. We also show that they achieve a better performance than the recently proposed cQA retrieval method.**

I. Introduction

Probabilistic language models are used to resolve various problems in information retrieval and text mining. In particular, latent Dirichlet allocation (LDA) [1] and its variants enable us to exploit the latent structure behind text. LDA was originally designed for texts represented by a bag of words, but its idea can also be applied to structured text. For example, the author-topic model utilizes the correlation between a given set of text and its author by using the latent topics [2]. Wang et al. proposed a topic model handling the co-occurrence of different kinds of texts [3].

Large corpora are important to obtain probabilistic models. In particular, the corpus size significantly affects the quality of the obtained probabilistic models. Various kinds of large corpora have recently become available. For example, Google provides the n-grams of the Web. Wikipedia is another large language resource for probabilistic language modeling.

Under these circumstances concerning probabilistic language models, one important research direction is how to combine multiple models obtained from different language resources. In this paper we discuss a method for combining multiple models for community-based Question answering (cQA) systems. Question answering (QA) is an important function in information retrieval in which users ask a question in natural language and retrieve the required information from a large amount of information sources, such as the Web. Although the Web may contain information that is relevant to a wide range of questions, it is hard to find the most relevant pages and synthesize them into a precise and concise answer.

Several cQA archives have recently been made available such as Yahoo! Answers.[1] In cQA archives, a user posts a question and other users answer it. These QAs are stored in the archive. Since the archive maintain a large amount of QA pairs, adequate answers to various questions are expected to be readily stored in the archive. Therefore, it is enough to just find the QAs most relevant to the question and thus, we can avoid the hard task of synthesizing the answers.

cQA retrieval has attracted a lot of attention from researchers. One of the key issues of cQA retrieval is the retrieval performance, just like the other IR tasks. For example, Xue et al. introduced a translation-based language model that overcomes the word-mismatch problem between a question and QA pair [4]. Wang et al. proposed to exploit the syntactical structure of questions and answers to improve the cQA performance [5].

We believe we need to take into consideration the matching between a question and QA pair from various aspects to improve the performance. Xue et al. [4] utilized two probabilistic models in addition to the translation model. The main concern of this paper is how to integrate various probabilistic models. The main contributions of this paper are:

- proposing three methods for combining probabilistic language models, and
- showing that the proposed methods improve the cQA retrieval using an evaluation corpus.

II. Problem Definition

At first we define the cQA problem and notations used in this paper. Recently, Various kinds of community-based information resources have recently been built on the Internet. Among them, cQA archives help us to obtain information for a wide range of problems. Suppose we want to get a new portable PC and are considering buying a notebook PC or an iPad and post the following question to a cQA system

> I am considering buying either an iPad or a notebook PC. The main purposes are for e-mail, listening to music, watching videos, and reading comics. Please tell me the advantages and disadvantages of the iPad.

[1] http://answers.yahoo.com/

1082-3409/10 $26.00 © 2010 IEEE
DOI 10.1109/ICTAI.2010.107

Many other people may have similar kinds of problems and have previously asked similar questions. In such cases, the cQA archives retain the answers, such as

> An iPad is lightweight and its touch panel is easy to use, but a notebook PC has more powerful functions than the iPad.

cQA systems usually provide search functions using keywords. However, there may be various kinds of answers related to the iPad in the archive such as, "where to buy" or "how to connect to the Internet", and they are returned as answers to the query based on the query word "iPad". As a result, the required answers are not always top-ranked in the returned answers.

For a query specified by a few sentences, the task of cQA retrieval is to retrieve answers to the query from the cQA archive and rank the answers according to relevance. The main differences from typical information retrieval are:

- queries are given by a few sentences instead of a couple of keywords,
- QA archive consists of question-answer pairs instead of single text, and
- relevance to the query is judged as to whether the retrieved answer contains the solution to the question.

cQA retrieval has recently attracted the attention of a lot of information retrieval researchers. For example, Mori et al. focused on patterns in questions and answers and proposed a QA retrieval method that exploits the correspondence between patterns which appear both in the questions and answers [6]. Jeon et al. used a probabilistic machine translation model to extract the co-relation between words in the questions and answers [7]. Xue et al. extended Jeon's method by incorporating a probabilistic language model [4]. Wang et al. proposed the use of the syntactical similarity between the parsing trees of questions [5]. Ko et al. proposed measuring the relevance of an answer to the question by combining several evidences of relevance by using the logistic regression [8], [9].

In this paper, we focus on the combination of multiple probabilistic models to improve the performance of the cQA retrieval. Although the proposed framework is similar to Xue's method, they manually integrate multiple models whereas we propose a learning method to combine them.

We list the notations used in this paper. For a set (resp. sequence) $\boldsymbol{s}$, $|\boldsymbol{s}|$ denotes the number of elements in (resp. the length of) $\boldsymbol{s}$. For a set or sequence $\boldsymbol{s}$ and a component c of $\boldsymbol{s}$, $N_c^{\boldsymbol{s}}$ denotes the number of times the component c appears in $\boldsymbol{s}$.

In this paper, we handle both the questions and answers as a bag of words just like in [1], [7], [4]. A question and answer are respectively denoted as $\boldsymbol{q} := \{q_i\}_{i=1}^{|\boldsymbol{q}|}$ and $\boldsymbol{a} := \{a_i\}_{i=1}^{|\boldsymbol{a}|}$ where q_i (resp. a_i) is a word appearing in the question (resp. answer). A cQA archive consists of pairs of questions and answers $C := \{(\boldsymbol{q}_i, \boldsymbol{a}_i)\}_{i=1}^{|C|}$. We denote the set of questions included in the archive as $Q := \{\boldsymbol{q}_i\}_{i=1}^{|Q|}$ and the set of answers in the archive as $A := \{\boldsymbol{a}_i\}_{i=1}^{|A|}$. For a query $\boldsymbol{q}$, the cQA retrieval problem is to find ranked question-answer pairs in C, where a higher ranked question-answer pair more likely contains the answer to $\boldsymbol{q}$.

In this paper, we use two kinds of probability distributions, a multinomial distribution $Multi(\boldsymbol{\lambda})$ and a Dirichlet distribution $\mathcal{DIR}(\boldsymbol{\alpha})$ as in LDA, where $\boldsymbol{\lambda}$ and $\boldsymbol{\alpha}$ respectively denote the parameters of the multinomial and Dirichlet distributions. Suppose these probability distributions are defined over a set of events $E = \{e_i\}_{i=1}^{|E|}$. For an event $e \in E$, λ_e (resp. α_e) denotes the component for e of $\boldsymbol{\lambda}$ (resp. $\boldsymbol{\alpha}$).

III. cQA Retrieval by Multiple Models

A. Framework of cQA Retrieval by Multiple Models

As defined in the previous section, the task of cQA retrieval is to rank QA pairs in the C for a given query. We assume that a query is generated from a query-answer pair by using a probabilistic model. Queries may contain various kinds of words. Some words are generally used for questions. Others are a specific word for the objective question. Therefore, we assume that there are multiple probabilistic models that generate questions. We refer to these probabilistic models as *component models*. The purpose of this paper is to develop a method to calculate the relevance between a query and query-answer pair by combining the component models.

Let $\boldsymbol{M} := \{m_1, m_2, \cdots, m_M\}$ be a set of component models that generate a query $\boldsymbol{r}$ from a question-answer pair ($\boldsymbol{q}$, $\boldsymbol{a}$). Let $\Pr(\boldsymbol{r} \mid (\boldsymbol{q}, \boldsymbol{a}); m)$ denote the probability that the model $m \in M$ generates the query $\boldsymbol{r}$ from the question-answer pair $(\boldsymbol{q}, \boldsymbol{a})$.

The problem is to derive a *combination function*

$$\mathcal{R}(\Pr(\boldsymbol{r} \mid (\boldsymbol{q}, \boldsymbol{a}); m_1), \cdots, \Pr(\boldsymbol{r} \mid (\boldsymbol{q}, \boldsymbol{a}); m_M)) \qquad (1)$$

that represents the relevance between the query $\boldsymbol{r}$ and the question-answer pair $(\boldsymbol{q}, \boldsymbol{a})$ as well as to find an effective set of component models. The QA pairs in an archive are ranked according to Eq.(1). We concentrate on the combination function in this paper.

To make the problem feasible, we assume that each word in a query is generated independently from the query pair just as in many language models [1], [2], [7], [4], i.e.,

$$\Pr(\boldsymbol{r} \mid (\boldsymbol{q}, \boldsymbol{a}); m) := \prod_{w \in \boldsymbol{r}} \Pr(w \mid (\boldsymbol{q}, \boldsymbol{a}); m) \qquad (2)$$

for any model $m \in M$.

B. Component Models

In our framework, we can use any probabilistic model as a component model. For example, we can use a probabilistic model based on the syntactical structure like the one proposed in [5]. We use the following four models in this paper.

First, the sentences used for the question and answers may be generated by different models. For example, the phrase "how to" appears more often in questions than in answers. So, we use a probabilistic model for translation between the question and answers in the same way as the study conducted by Xue et al. [4]. We use the IBM model 1 [10] as the translation model. Let W and V be the sets of words in

two different languages L_1 and L_2, respectively. For any pair $w \in W$ and $v \in V$ of words, the IBM model defines the translation probability $\Pr(w \mid v)$.

The translation probability is estimated from a parallel corpus consisting of sentences in both languages that represent the same meaning. Suppose a parallel corpus $\{(\boldsymbol{w}_1, \boldsymbol{v}_1), (\boldsymbol{w}_2, \boldsymbol{v}_2), \cdots, (\boldsymbol{w}_n, \boldsymbol{v}_n)\}$ is given as training data, where $\boldsymbol{w}_i$ and $\boldsymbol{v}_i$ are the sentences from L_1 and L_2, respectively. Then, the translation probability is estimated by iteratively updating the probabilities by using the following formula

$$\Pr^{t+1}(w \mid v) \propto \sum_{i=1}^{n} c(w \mid v; \boldsymbol{w}_i, \boldsymbol{v}_i)$$

where

$$c(w \mid v; \boldsymbol{w}_i, \boldsymbol{v}_i) = \frac{\Pr^t(w \mid v)}{\sum_{j=1}^{|\boldsymbol{v}_i|} \Pr^t(w \mid v_{ij})} N_v^{\boldsymbol{v}_i} N_w^{\boldsymbol{w}_i} \ .$$

In these formulas, $\Pr^t$ (resp. $\Pr^{t+1}$) is the estimated probability at the tth (resp. $(t+1)$th) iteration. Note that $N_v^{\boldsymbol{v}_i}$ denotes the number of occurrence of v in $\boldsymbol{v}_i$.

We use four component models in this paper that are defined by using Eqs. (3), (4), (5), and (7). The first model M_{ml} is the query likelihood language model by using the maximum likelihood estimation. It is defined by

$$\Pr(w \mid (\boldsymbol{q}, \boldsymbol{a}); M_{ml}) := \frac{N_w^{\boldsymbol{q}}}{|\boldsymbol{q}|} \ . \tag{3}$$

This model is frequently used in information retrieval [11]. We refer to this model as a *language model*. Note that we use the question part of the QA pair because the word occurrence probability of the answers may be different from that of the questions.

The remaining three models are introduced to handle words that appear in the query $\boldsymbol{r}$, but that do not appear in the query $\boldsymbol{q}$ because the language model cannot handle these words. The second model M_{tr} is the translation model estimated from the parallel corpus

$$\{(\boldsymbol{q}_1, \boldsymbol{a}_1), (\boldsymbol{q}_2, \boldsymbol{a}_2), \cdots, (\boldsymbol{q}_{|C|}, \boldsymbol{a}_{|C|})\} \ ,$$

where C denotes the set of QA pairs in the cQA archive. Let $\Pr(w \mid v, C)$ be the resultant translation probability when using IBM model I described above. The model M_{tr} is defined as

$$\Pr(w \mid (\boldsymbol{q}, \boldsymbol{a}); M_{qatr}) := \frac{1}{|\boldsymbol{a}|} \sum_{t \in \boldsymbol{a}} \Pr(w \mid t, C) \ . \tag{4}$$

We refer to this model as a *QA translation model*.

Similarly, we can take into consideration a translation model between questions. Questions are generally short whereas answers are long. So, we first translate a word w in a question into a word v in the answer and then translate v into a word w' in the question. To make this work, we obtain the translation model between the questions by estimating the model using the parallel corpus

$$\{(\boldsymbol{q}_1, \boldsymbol{a}_1), (\boldsymbol{q}_2, \boldsymbol{a}_2), \cdots, (\boldsymbol{q}_{|C|}, \boldsymbol{a}_{|C|}), (\boldsymbol{a}_1, \boldsymbol{q}_1), (\boldsymbol{a}_2, \boldsymbol{q}_2), \cdots, (\boldsymbol{a}_{|C|}, \boldsymbol{q}_{|C|})\} \ .$$

Let $\Pr(w \mid v, CC^{-1})$ be the resultant translation probability. Then, the model M_{qqtr} is defined as

$$\Pr(w \mid (\boldsymbol{q}, \boldsymbol{a}); M_{qqtr}) := \frac{1}{|\boldsymbol{q}|} \sum_{t \in \boldsymbol{q}} \Pr(w \mid t; CC^{-1}) \ . \tag{5}$$

We refer to this model as a *QQ translation model*.

Finally, we introduce a corpus-wide distribution of words smoothed by using the Good-Turing estimation [12]. For a frequency f of a word in the question-answering archive C, let W_f denote the number of words whose frequency is f. The Good-Turing estimator uses the following modified frequency

$$\hat{f} := (f+1)\frac{|W_{f+1}|}{|W_f|} \ .$$

For a word w, let N_w^C denote the frequency of w in the question-answering archive C. The probability that a word w occurs is given by

$$\Pr(w \mid C) := \begin{cases} \frac{\hat{N}_w^C}{|W|} & N_w^C > 0 \\ \frac{|W_1|}{|W_0| \cdot |W|} & N_w^C = 0 \end{cases} \ . \tag{6}$$

We use the Good-Turing estimator as a corpus-wide smoother:

$$\Pr(w \mid (\boldsymbol{q}, \boldsymbol{a}); M_{bg}) := \Pr(w \mid C) \ . \tag{7}$$

We refer to this model as a *background model* and denote it as M_{bg}. Note that this model gives the probability of a word w independent of a question-answering pair $(\boldsymbol{q}, \boldsymbol{a})$.

IV. Combination of Component Models

A. Finite Mixture Model

This section derives two models for combining the component models. The first model is a mixture of the component models. Let M be a set $\{m_i\}_i$ of component models and $\Pr(w \mid (\boldsymbol{q}, \boldsymbol{a}); m)$ denote the probability that the word w is generated by a model $m \in M$. Then, the word generation probability is described as

$$\Pr(w \mid (\boldsymbol{q}, \boldsymbol{a})) := \sum_i \lambda_i \Pr(w \mid (\boldsymbol{q}, \boldsymbol{a}); m_i) \ , \tag{8}$$

where $\boldsymbol{\lambda} := (\lambda_i)_i$ is mixture proportion, i.e., $\sum_i \lambda_i = 1$ holds.

We regard the mixture proportion $\boldsymbol{\lambda}$ as a multinomial probability. The first model is a direct application of LDA. By regarding the component model for each word in a question as a latent topic, we can build the following generative model in the same way as LDA. Let T denote a set of triplets $(\boldsymbol{r}, \boldsymbol{q}, \boldsymbol{a})$, where $\boldsymbol{r}$ is a query whereas $(\boldsymbol{q}, \boldsymbol{a})$ is the corresponding QA pair in the archive. For a given Dirichlet parameter $\boldsymbol{\alpha}$, the generative model is as follows:

Generative Mixture Model I
for all $(\boldsymbol{r}, \boldsymbol{q}, \boldsymbol{a})$ in T

1) generate a multinomial distribution $Multi(\boldsymbol{\lambda}) \sim \mathcal{DIR}(\boldsymbol{\alpha})$
2) for each word $w \in \boldsymbol{r}$
   a) choose a model $m \in M \sim Multi(\boldsymbol{\lambda})$ and
   b) generate $w \sim \Pr(w; m)$.

Figure 1 (a) shows a graphical model of generative mixture model I. As shown in the generative procedure, one model mixture proportion $\boldsymbol{\lambda}_i$ is generated for each query-question-answer triplet in the training data. In the following discussion, we abbreviate $\Pr(w \mid (\boldsymbol{q}, \boldsymbol{a}); m)$ to $\Pr(w; m)$ when the QA pair is obvious in the context.

We can estimate the multinomial distributions $\boldsymbol{\Lambda} := \{\boldsymbol{\lambda}_1, \cdots, \boldsymbol{\lambda}_{|T|}\}$ where $\boldsymbol{\lambda}_i := (\lambda_{im})_{m \in M}$ $(1 \leq i \leq |T|)$ by using the Gibbs sampling, just as in [13]. Let $\boldsymbol{m} := (m_{ij})_{ij}$ denote a model assignment to each word in the questions in T, where m_{ij} is the model assignment to the jth word in the question of ith query-question-answer triplet in T. Then, the complete-data likelihood is given by

$$\begin{aligned}
&\Pr(T, \boldsymbol{m}, \boldsymbol{\Lambda}; \boldsymbol{\alpha}) \\
&= \prod_{i=1}^{|T|} \Pr(\boldsymbol{\lambda}_i; \boldsymbol{\alpha}) \left[\prod_{j=1}^{|\boldsymbol{r}_i|} \Pr(m_{ij} \mid \boldsymbol{\lambda}_i) \Pr(r_{ij}; m_{ij}) \right] \\
&= \prod_{i=1}^{|T|} \left[\mathcal{D}(\boldsymbol{\alpha}) \prod_{m \in M} \lambda_{im}^{N_m^{\boldsymbol{m}_i} + \alpha_m - 1} \right] \left[\prod_j \Pr(r_{ij}; m_{ij}) \right] \quad (9)
\end{aligned}$$

where $N_m^{\boldsymbol{m}_i}$ denotes the number of times that a model m is assigned to the words in $\boldsymbol{r}_i$, whereas $\mathcal{D}(\boldsymbol{\alpha})$ is the normalizing coefficient of the Dirichlet distribution, i.e.,

$$\mathcal{D}(\boldsymbol{\alpha}) := \frac{\Gamma(\sum_k \alpha_k)}{\prod_k \Gamma(\alpha_k)} \ .$$

By marginalizing Eq. (9) by using the multinomial parameters $\boldsymbol{\Lambda}$, we obtain

$$\begin{aligned}
\Pr(T, \boldsymbol{m}; \boldsymbol{\alpha}) &= \int \Pr(T, \boldsymbol{m}, \boldsymbol{\Lambda}; \boldsymbol{\alpha}) \ d\boldsymbol{\Lambda} \\
&= \prod_{i=1}^{|T|} \frac{\Gamma(\sum_m \alpha_m)}{\prod_m \Gamma(\alpha_m)} \frac{\prod_m \Gamma(N_m^{\boldsymbol{m}_i} + \alpha_m)}{\Gamma(\sum_m N_m^{\boldsymbol{m}_i} + \alpha_m)} \prod_j \Pr(r_{ij}; m_{ij}) \ . \quad (10)
\end{aligned}$$

Let $\boldsymbol{m}_{-ij}$ denote the model assignment $\boldsymbol{m}$ except for the assignment to the jth word of the ith query $\boldsymbol{r}_i$. From Eq. (10), the full conditional probability for the Gibbs sampling of this model is given by

$$\begin{aligned}
&\Pr(m_{ij} = m \mid T, \boldsymbol{m}_{-ij}; \boldsymbol{\alpha}) \\
&\propto \frac{N_m^{\boldsymbol{m}_i^{-ij}} + \alpha_m}{\sum_{m'} (N_{m'}^{\boldsymbol{m}_i^{-ij}} + \alpha_{m'})} \Pr(r_{ij}; m) \ , \quad (11)
\end{aligned}$$

where $\boldsymbol{m}_i^{-ij}$ denotes the model assignment to the ith query $\boldsymbol{r}_i$ except for the jth word in $\boldsymbol{r}_i$. This formula is used to reassign a model to each word r_{ij} in the Gibbs sampling. After convergence, we obtain a mixture proportion as

$$\lambda_{im} = \frac{N_m^{\boldsymbol{m}_i^{-ij}} + \alpha_m}{\sum_{m'} (N_{m'}^{\boldsymbol{m}_i^{-ij}} + \alpha_{m'})} \ , \quad (12)$$

for $(1 \leq i \leq |T|, m \in M)$. Algorithm 1 presents the procedural outline.

Algorithm 1 Estimate the Generative Mixture Model I

```
repeat
  set initial models to each word in T
  for all (r, q, a) in T do
    for all w ∈ r do
      re-assign model randomly according to the probability distribution Eq. (11)
    end for
  end for
until convergence
```

The drawback of mixture model I is that the mixture proportion $\boldsymbol{\lambda}_i$ of each query-question-answer triplet is independently estimated. This is recognized in Eq. (11) for the Gibbs sampling, where the number of model m assignments is examined in $\boldsymbol{m}_i^{-ij}$, i.e., it is within the query-question-answer topic assignment. In other words, the corpus-wide information is not utilized in this model. In LDA, the corpus-wide information is utilized in the word generation process from each latent topic, although the corresponding part in the mixture model I is separately estimated in each component model.

To overcome this drawback, we use the same mixture proportion $\boldsymbol{\lambda}$ for all query-question-answer triplets. For a given Dirichlet parameter $\boldsymbol{\alpha}$, the generative model is as follows:

Generative Mixture Model II

1) generate a multinomial distribution $Multi(\boldsymbol{\lambda}) \sim \mathcal{DIR}(\boldsymbol{\alpha})$
2) for each word $w \in \boldsymbol{r}$ in each $(\boldsymbol{r}, \boldsymbol{q}, \boldsymbol{a})$ in T
   a) choose a model $m \in M \sim Multi(\boldsymbol{\lambda})$ and
   b) generate a word according to $\Pr(w; m)$.

Figure 1 (b) shows a graphical model of generative mixture model II. As shown in the figure, the single model mixture proportion is used for all the query-question-answer triplets in this model although one model mixture proportion per triplet is used in the generative mixture model I.

The complete-data likelihood for model mixture model II is given by

$$\begin{aligned}
&\Pr(T, \boldsymbol{m}, \boldsymbol{\lambda}; \boldsymbol{\alpha}) \\
&= \Pr(\boldsymbol{\lambda}; \boldsymbol{\alpha}) \prod_{i=1}^{|T|} \prod_{j=1}^{|\boldsymbol{r}_i|} \Pr(m_{ij} \mid \boldsymbol{\lambda}) \Pr(r_{ij}; m_{ij}) \ , \quad (13)
\end{aligned}$$

and the marginalized complete data likelihood is

$$\begin{aligned}
\Pr(T, \boldsymbol{m}; \boldsymbol{\alpha}) &= \int \Pr(T, \boldsymbol{m}, \boldsymbol{\lambda}; \boldsymbol{\alpha}) \ d\boldsymbol{\lambda} \\
&= \frac{\Gamma(\sum_m \alpha_m)}{\prod_m \Gamma(\alpha_m)} \frac{\prod_m \Gamma(N_m^{\boldsymbol{m}} + \alpha_m)}{\Gamma(\sum_m N_m^{\boldsymbol{m}} + \alpha_m)} \prod_{i=1}^{|T|} \prod_{j=1}^{|\boldsymbol{r}_i|} \Pr(r_{ij}; m_{ij}) \ , \quad (14)
\end{aligned}$$

where $N_m^{\boldsymbol{m}}$ denotes the number of times a model m is assigned to the words in T. Then, the full conditional probability for

the Gibbs sampling of this model is given by

$$\Pr(m_{ij} = m \mid T, \boldsymbol{m}^{-ij}; \boldsymbol{\alpha}) \propto \frac{N_m^{\boldsymbol{m}^{-ij}} + \alpha_m}{\sum_{m'}(N_{m'}^{\boldsymbol{m}^{-ij}} + \alpha_{m'})} \Pr(r_{ij}; m) \ . \quad (15)$$

Note that the full conditional probability is given by using a corpus-wide model assignment $\boldsymbol{m}^{-ij}$ although it is calculated within a query-question-answer triplet $\boldsymbol{m}_i^{-ij}$ in mixture model I as in Eq. (11).

After convergence of the Gibbs sampling, we obtain the following mixture proportion.

$$\lambda_m = \frac{N_m^{\boldsymbol{m}} + \alpha_m}{\sum_{m'}(N_{m'}^{\boldsymbol{m}} + \alpha_{m'})} \quad (16)$$

for each $m \in M$.

The procedure for the generative mixture model is the same as for Algorithm 1 for the generative mixture model I except for the use of Eq. (15) for the model re-assignment instead of Eq. (11).

B. Likelihood Ratio Model

This section proposes a new model called a *likelihood ratio model*. This model modifies the mixture models proposed in the previous section in two ways.

We used one mixture proportion for each query-question-answer triplet in mixture model I, whereas we used single mixture proportions for all the query-question-answer triplets in mixture model II. For the first modification, the likelihood ratio model uses several mixture proportions. First, let us divide the word set W into mutually distinctive subsets. We refer to these subsets as a *word cluster*. Although there are many ways to divide the word set, we currently divide a set of words in the following way. First, we prepare the word clusters $C_1, C_2, \cdots, C_{|M|}$, where each cluster C_i corresponds to a component model $m \in M$. Then, each word w is categorized into C_m where

$$m =: \operatorname*{argmax}_{m \in M} \Pr(w; m) \ .$$

We took into consideration the following generative model.

Word Cluster Model

1) for each word cluster $c \in C$
   a) choose a multinomial distribution $Multi(\boldsymbol{\lambda}_c) \sim \mathcal{DIR}(\boldsymbol{\alpha})$ for model selection
2) for each word w in the query $\boldsymbol{r}$ in each triplet $(\boldsymbol{r}, \boldsymbol{q}, \boldsymbol{a})$ in T
   a) choose a model $m \in M \sim Multi(\boldsymbol{\lambda}_{\mathcal{C}(w)})$,
   b) generate a word $w \sim \Pr(w; m)$,

where $\mathcal{C}(w)$ denotes the class to which a word w belongs. Figure 1 shows a graphical model of the word class generative model. As shown in the figure, one model mixture proportion is generated for each word cluster. Usually, there are fewer word clusters than the number of query-question-answer triplets. Therefore, the word cluster model is less complex than generative mixture model I, but it is more complex than generative mixture model II. By tuning the number of word clusters, we can make an adequately complex model.

The complete-data likelihood of the word cluster model is given by

$$\Pr(T, \boldsymbol{m}, \boldsymbol{\Lambda}; \boldsymbol{\alpha}) = \prod_{c \in C} \Pr(\boldsymbol{\lambda}_c; \boldsymbol{\alpha}) \prod_{i=1}^{|T|} \prod_{j=1}^{|\boldsymbol{r}_i|} \Pr(m_{ij} \mid \boldsymbol{\lambda}_{\mathcal{C}(r_{ij})}) \Pr(r_{ij}; m_{ij}), \quad (17)$$

where $\boldsymbol{m}$ denotes the model assignment to words included in the queries in T, whereas $\boldsymbol{\Lambda}$ denotes the set $\{\boldsymbol{\lambda}_c\}_{c \in C}$ of mixture proportions of word classes.

The marginalized complete data likelihood is

$$\begin{aligned} \Pr(T, \boldsymbol{m}; \boldsymbol{\alpha}) &= \int \Pr(T, \boldsymbol{m}, \boldsymbol{\Lambda}; \boldsymbol{\alpha}) \ d\boldsymbol{\Lambda} \\ &= \left[\prod_{c \in C} \frac{\Gamma(\sum_m \alpha_m)}{\prod_m \Gamma(\alpha_m)} \frac{\prod_m \Gamma(N_{cm}^{\boldsymbol{m}} + \alpha_m)}{\Gamma(\sum_m N_{cm}^{\boldsymbol{m}} + \alpha_m)} \right] \left[\prod_{i=1}^{|T|} \prod_{j=1}^{|\boldsymbol{r}_i|} \Pr(r_{ij}; m_{ij}) \right] \ . \end{aligned} \quad (18)$$

Then, the full conditional probability for the model selection is given by

$$\Pr(m_{ij} = m \mid T, \boldsymbol{m}^{-ij}; \boldsymbol{\beta}) \propto \frac{N_{\mathcal{C}(r_{ij})m}^{\boldsymbol{m}^{-ij}} + \alpha_m}{\sum_{m'}(N_{\mathcal{C}(r_{ij})m'}^{\boldsymbol{m}^{-ij}} + \alpha_{m'})} \Pr(r_{ij}; m) \ . \quad (19)$$

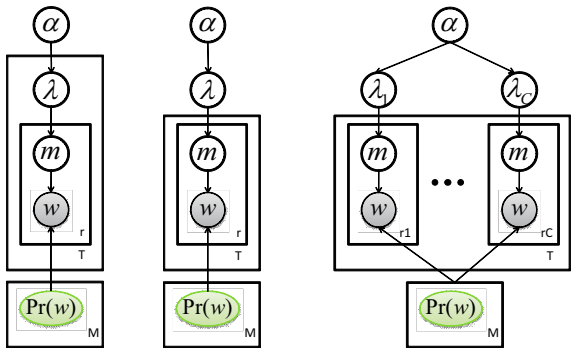
The procedure for the word cluster model is the same as that for Algorithm 1 for generative mixture model I, except for the use of Eq. (19) for the model re-assignment instead of Eq. (11).

After convergence of Gibbs sampling, we obtain a set of mixture proportions as

$$\lambda_{cm} = \frac{N_{cm}^{\boldsymbol{m}} + \alpha_m}{\sum_{m'}(N_{cm'}^{\boldsymbol{m}} + \alpha_{m'})}, \quad (20)$$

for each model $m \in M$ and class $c \in C$.

The background model introduced in Sec. III-B was used as a smoother in the mixture models. In information retrieval, we usually detect important words that have much information in order to discriminate the objective text from the others and give more weight to these discriminative words when calculating the similarity between documents. As for the second modification, we give more weight to these discriminative words. Let $M_{bg} \subset M$ be the set of background models in the model set M. We refer to the remaining models $M_{fg} := M - M_{bg}$ as *foreground models*. We obtain the model mixture proportions of the word cluster model for both the foreground and background models. Let $\boldsymbol{\Lambda} := \{\boldsymbol{\lambda}_c\}_{c \in C}$ and



(a) Model I (b) Model II (c) Likelihood Ratio Model

Fig. 1. Graphical models for combining component models: Symbols M, T, r, and C, respectively, denote the numbers of models, training query-question-answer triplets, words included in a query, and word clusters. The symbol r_i $(1 \leq i \leq C)$ in the rightmost graphical model denotes the number of times a word appearing in a query belongs to a word cluster c_i. The shaded circles denote the observed data, whereas the green circles denote the probabilities estimated outside the generative model.

$\boldsymbol{\Theta} := \{\boldsymbol{\theta}_c\}_{c \in C}$ be the mixture proportions for the foreground and background models, respectively.

We assume that a word is discriminative if its likelihood in the foreground models differs from the likelihood in the background models. According to this assumption, we calculate the score of a word w in a query $\boldsymbol{r}$ for a QA pair $(\boldsymbol{q}, \boldsymbol{a})$ by

$$S(w \mid (\boldsymbol{q}, \boldsymbol{a})) := \frac{\sum_{m \in M_{fg}} \theta_{\mathcal{C}(w)m} \Pr(w_i; m)}{\sum_{m' \in M_{bg}} \lambda_{\mathcal{C}(w)m'} \Pr(w_i; m')} , \qquad (21)$$

using the mixture proportions $\boldsymbol{\Theta}$ and $\boldsymbol{\Lambda}$. Compared to Eq. (8) for the score of the mixture models, the likelihood ratio model uses the difference in likelihood between the foreground and background models as the word weight.

V. Experimental Results

A. Data Set

In this experiment we used Japanese Yahoo! answers [2]. It contains about three million QA pairs. We chose pairs in the Internet category. The total number of used QA pairs was 171,816. Since the Japanese language has no explicit word boundary, we applied the morphological analyzer MeCab [3] to extract words and then removed the stop words. As a result, the questions and answers are represented by a bag of words.

We built an evaluation corpus for the cQA retrieval. First, we randomly chose 32 questions from the QA pairs. Then, we gathered candidate answers by pooling. In this process, we used the following six QA retrieval methods:

- the Language Model defined by Eq. (3),
- Okapi/BM25 (see below),
- Xue's method [4],
- proposed mixture model I with symmetric Dirichlet parameters, i.e., $\alpha_1 = \alpha_2 = \cdots = \alpha_{|M|}$
- proposed mixture model I with manually tuned Dirichlet parameters, and
- proposed mixture model II.

BM25[14] is frequently used in information retrieval. For a query $\boldsymbol{r}$ and QA pair $\boldsymbol{d} := \boldsymbol{q} \cup \boldsymbol{a}$, it is defined as

$$BM25(\boldsymbol{r}, (\boldsymbol{q}, \boldsymbol{a})) := \sum_{w \in \boldsymbol{r}} IDF(w) TF(d, \boldsymbol{d}) \frac{(k_3 + 1)\, N_w^{\boldsymbol{r}}}{(k_3 + N_w^{\boldsymbol{r}})} , \qquad (22)$$

where

$$IDF(w) := log \frac{|T| - df(w) + 0.5}{df(w) + 0.5} ,$$

$$TF(w) := \frac{N_w^{\boldsymbol{d}}(k_1 + 1)}{N_w^{\boldsymbol{d}} + k_1(1 - b + b\frac{|\boldsymbol{d}|}{avgdl})} .$$

The function $df(w)$ denotes the document frequency, i.e., the number of documents that contain the word w. On the other hand, $N_w^{\boldsymbol{r}}$ denotes the number of times a word w appears in a query $\boldsymbol{r}$.

We applied these six retrieval methods for each query and obtained six sets of ranked QA pairs. Then, we chose the top-50 ranked QAs from each set and merged them to create the candidate answers. Finally, four graduate students manually judged the relevance of each candidate to obtain the relevant QA pairs for each query.

B. Parameter Estimation

The parameters of the models were estimated from the QA pairs, except for those relevant to the 32 queries. We implemented the Gibbs sampler ourselves, whereas we used GIZA++ toolkit [15] [4] for the parameter estimation and likelihood calculation of the translation model.

C. Evaluation Metrics

We evaluated the performances of the QA methods by observing the precision for the top-K ranked answers (P@K) and the average precision (MAP) [16]. The P@K is frequently used for evaluating information retrieval methods when it is difficult to enumerate all the relevant documents to each query. Suppose we obtain a ranked document $(d_1, d_2, \cdots, d_k)$ by using a retrieval method for a query and its relevance is $(x_1, x_2, \cdots, x_k)$, where

$$x_i = \begin{cases} 1 & \text{if } d_i \text{ is relevant to the query} \\ 0 & \text{otherwise} \end{cases} .$$

Then, the P@K for the query is defined as

$$P@K := \frac{1}{K} \sum_{i=1}^{K} x_i .$$

In the following discussion, we use the average P@10 for the above mentioned 32 queries.

[2] http://chiebukuro.yahoo.co.jp/

[3] http://mecab.sourceforge.net

[4] http://www.fjoch.com/GIZA++.html

$\sum_k \alpha_k$	P@10	MAP
60k	0.372	0.377
600k	0.378	0.404
6M	**0.381**	**0.407**
60M	0.378	0.404

TABLE I
PERFORMANCE COMPARISON W.R.T. DIRICHLET PARAMETERS

	P@10	MAP
LM	0.267	0.281
BM25	0.284	0.311
Xue[4]	0.363	0.363
Mixture I	0.3625	0.4056
Mixture II	0.381	0.407
LRatio	**0.392**	**0.412**

TABLE II
PERFORMANCE COMPARISON

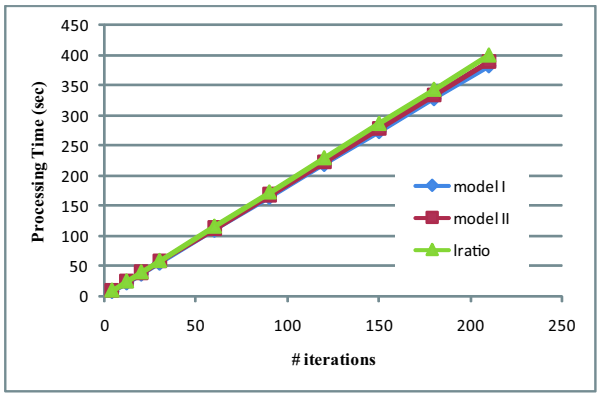


Fig. 2. Processing Time. "model I", "model II", and "lratio" respectively plot the processing times with respect to the number of iterations in Gibbs sampling.

MAP is also frequently used to evaluate information retrieval. It is defined as

$$MAP := \frac{1}{\sum_{i=1}^{k}} \sum_{i=1}^{k} \frac{x_i}{i}(1 + \sum_{j=1}^{i-1} x_j) \ .$$

We use the average MAP for the above mentioned 32 queries.

D. Performance Evaluation

First, we evaluated the effect of the Dirichlet parameter estimation for mixture model II. In this experiment, we manually tuned the parameters and compared the performance to the symmetric Dirichlet distributions, i.e., all the parameter values are equivalent. Table I lists the performance for various parameter values. As shown in the table, both P@10 and MAP are slightly affected by the Dirichlet parameters.

Next, we compared the proposed methods with the following three methods:

- the language model (LM) defined by Eq. (3),
- Okapi/BM25 (BM25) defined by Eq. (22), and
- Xue's method (Xue) [4] that was implemented by ourselves.

Both LM and MB25 are basic information retrieval methods and we adopted them as our baseline methods. In addition, we compared the proposed methods with Xue's method because it was recently proposed for use as a cQA retrieval method and it consists of mixture models, although the mixture proportions are manually tuned.

Table II lists the performance in terms of the MAP and P@10. The proposed methods are denoted as *Mixture I*, *Mixture II*, and *LRatio*.

First, both the proposed and Xue's methods significantly outperform the baseline methods. The paired T-test indicates that both methods are superior to the baseline methods with a probability of more than 95% on both P@10 and MAP. Second, the proposed method improves Xue's method. The paired T-test indicates the proposed methods are superior to Xue's with a probability of more than 95% on MAP, although the statistical significance is not shown on P@10. This result indicates that the proposed methods estimate a good mixture proportion of the component models and effectively incorporates the translation models as well as the background model. Finally, the *LRatio* achieved the best performance amongst the three proposed methods.

E. Processing Efficiency

In this section, we show the processing efficiency of the Gibbs sampling algorithms for the three proposed models. We conducted all the experiments on a PC with Intel(R) Xeon(R) W5590 3.33 GHz processors and 32 GB RAM. We implemented our algorithms in C++ and compiled with g++ (GCC) 4.3.2 on Debian GNU/Linux.

The processing time for estimating the mixture proportion of component models is $O(lt)$ per iteration, where l and t respectively denote the total length of questions included in the training data and the number of topics, i.e., the number of component models. Compared to LDA, the number of topics is much less than the one used in ordinary problems, the required processing time is less than those applications. Another factor of the processing efficiency is the number of iterations until convergence. For all the three models, the parameters converged at about 200 iterations. Fig. 2 shows the average processing time with respect to the number of iterations of Gibbs sampling. As shown in this graph, no significant difference in processing time was observed among three models. The processing time for estimating the mixture proportions were about 400 seconds. These results show that the proposed method is efficient enough for practical use.

VI. CONCLUSION

This paper proposed three models for combining multiple probabilistic language models for cQA retrieval and their parameter estimation methods. Although the proposed mixture models are regarded as an extension of the one proposed by Xue el al.[4], we focused on a method to combine the component probabilistic language models in this study. We

experimentally showed that the proposed methods effectively combine multiple models for cQA retrieval. The proposed methods significantly outperform the widely used language models and Okapi BM25. We also showed that the proposed methods performed better than in [4].

We currently use four component models. One future research direction is to incorporate other models and check whether the proposed combination methods work effectively for larger numbers of component models. As for word clusters, we currently define the clusters in an ad hoc manner. There are several methods for creating word clusters. Therefore, another future technical problem is to check the effect of a clustering method on the performance of the proposed likelihood ratio model. In the experiments, we used cQAs in the Internet domain due to the hardness of building an evaluation data set. We plan to evaluate the proposed method by using cQAs in other genres. There are several cQA archives open to public. Another future work is to apply the proposed method to these cQA archives. We applied the proposed method in this paper to Japanese cQA archives. Our concern is to check the effectiveness of the proposed method for cQAs in other languages.

ACKNOWLEDGMENT

This research was partly supported by Grant-in-Aid for sientific Research on Priority Areas "Infoplosion" (18049069) and Grant-in-Aid for sientific Research (B) (20300038).

REFERENCES

[1] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.

[2] M. Rosen-Zvi, T. Griffiths, M. Steyvers, and P. Smyth, "The author-topic model for authors and documents," in *20th Annual Conference on Uncertainty in Artificial Intelligence*, 2004, pp. 487–494.

[3] X. Wang, N. Mohanty, and A. McCallum, "Group and topic discovery from relations and text," in *Proc. 3rd Intl. Workshop on Link Discovery (LinkKDD-2005)*, 2005, pp. 28–35.

[4] X. Xue, J. Jeon, and W. B. Croft, "Retrieval models for question and answer archives," in *Proc. Annual International ACM SIGIR Conference on Research and development in information retrieval (SIGIR2008)*, 2008, pp. 475–482.

[5] K. Wang, Z. Ming, and T.-S. Chua, "A syntactic tree matching approach to finding similar questions in community-based qa services," in *Proc. Annual International ACM SIGIR Conference on Research and development in information retrieval (SIGIR2009)*, 2009, pp. 187–194.

[6] T. Mori, M. Sato, and M. Ishioroshi, "Answering any class of japanese non-factoid question by using the web and example Q&A pairs from a social Q&A website," in *Proc. Web Intelligence (WI-2008)*, 2008, pp. 59–65.

[7] J. Jeon, W. B. Croft, and J. H. Lee, "Finding similar questions in large question and answer archives," in *CIKM*, 2005, pp. 84–90.

[8] J. Ko, L. Si, and E. Nyberg, "A probabilistic framework for answer selection in question answering," in *Proc. Human Language Technology Conference (NAACL-HLT 2007)*, 2007.

[9] ——, "A probabilistic graphical model for joint answer ranking in question answering," in *Proc. Annual International ACM SIGIR Conference on Research and development in information retrieval (SIGIR2007)*, 2007, pp. 343–350.

[10] P. Brown, V. Della Pietra, S. Della Pietra, and R. Mercer, "The mathematics of statistical machine translation: Parameter estimation," *Computational linguistics*, vol. 19, no. 2, pp. 263–311, 1993.

[11] J. M. Ponte and W. B. Croft, "A language modeling approach to information retrieval," in *Proc. Annual International ACM SIGIR Conference on Research and development in information retrieval (SIGIR1998)*, 1998, pp. 275–281.

[12] I. J. Good, "The population frequencies of species and the estimation of population parameters," *Biometrika*, vol. 40, pp. 237–264, 1953.

[13] T. L. Griffiths and M. Steyvers, "Finding scientific topics," in *Proc. of the National Academy of Sciences, 101 (suppl. 1)*, 2004, pp. 5228–5235.

[14] K. Sparck Jones, S. Walker, and S. Robertson, "A probabilistic model of information retrieval: development and comparative experiments," *Information Processing and Management*, vol. 36, no. 6, pp. 809–840, 2000.

[15] F. J. Och and H. Ney, "A systematic comparison of various statistical alignment models," *Computational Linguistics*, vol. 29, no. 1, pp. 19–51, 2003.

[16] C. Buckley and E. Voorhees, "Evaluating evaluation measure atability," in *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*. ACM New York, NY, USA, 2000, pp. 33–40.