Hindawi Publishing Corporation
ISRN Artificial Intelligence
Volume 2013, Article ID 829630, 17 pages
http://dx.doi.org/10.1155/2013/829630

Research Article

Gamma-Poisson Distribution Model for Text Categorization

Hiroshi Ogura, Hiromi Amano, and Masato Kondo

Department of Information Science, Faculty of Arts and Sciences, Showa University, 4562 Kamiyoshida, Fujiyoshida City, Yamanashi 403-0005, Japan

Correspondence should be addressed to Hiroshi Ogura; ogura@cas.showa-u.ac.jp

Received 29 January 2013; Accepted 4 March 2013

Academic Editors: K. W. Chau, C. Chen, G. L. Foresti, and M. Loog

We introduce a new model for describing word frequency distributions in documents for automatic text classification tasks. In the model, the gamma-Poisson probability distribution is used to achieve better text modeling. The framework of the modeling and its application to text categorization are demonstrated with practical techniques for parameter estimation and vector normalization. To investigate the efficiency of our model, text categorization experiments were performed on 20 Newsgroups, Reuters-21578, Industry Sector, and TechTC-100 datasets. The results show that the model allows performance comparable to that of the support vector machine and clearly exceeding that of the multinomial model and the Dirichlet-multinomial model. The time complexity of the proposed classifier and its advantage in practical applications are also discussed.

1. Introduction

The Poisson distribution is one of the most commonly used models for describing the number of random occurrences of a phenomenon in a specified unit of space or time. This means that if we want to model the number of discrete occurrences that take place during a given length, we should first check whether or not the Poisson distribution provides a good approximation. For text modeling, it is justified to adopt the Poisson distribution for describing the number of occurrences of a certain word in documents of fixed length when the independent assumption of each word occurrence holds in an approximate sense. It has been well established, however, that the Poisson model does not fit observed data [1, 2]. The reason for the failure of the Poisson model is that, for most words, the predicted variance, which is equal to the Poisson mean (the expected number of occurrences during the given interval), systematically underestimates the actual variance. Although this inadequate description of words distribution with the Poisson model can be used for key words selection in information retrieval [1] and for feature selection in text categorization [2–4] improvement in the accuracy of description is inevitably needed in order to build a high-performance text classifier using the model.

As has been proposed by Church and Gale [5], it is natural to extend the simple Poisson to a Poisson mixture in order to describe the observed variance in actual documents. Here, a Poisson mixture is a probability density function that is expressed as a sum of infinite Poisson distributions with a certain weighting function. The Poisson mixture is therefore considered to be a hierarchical model because a two-step process is required to generate a sample; to get a sample, we first pick a Poisson distribution with a certain probability according to the weighting function and then pick the sample from the chosen Poisson distribution. A reasonable choice of the weighting function is the conjugate prior of the Poisson, that is, the gamma distribution, because it greatly simplifies mathematical treatments [6], and this choice leads to the joint gamma-Poisson distribution.

In this paper, we focus on the use of the gamma-Poisson distribution to construct a new generative probabilistic text classifier. We believe that this is worthwhile for the following reasons.

(i) Several attempts to extend the original generative probabilistic classifier by using a conjugate prior for better text modeling have already been suggested; the reported conjugate prior-likelihood pairs are the Dirichlet-multinomial [7], the beta-negative binomial [8], and the beta-binomial [9]. To the best of our knowledge, the gamma-Poisson distribution has not yet been used to construct a generative

probabilistic text classifier. As mentioned above, since the model using the gamma-Poisson distribution can be regarded as one of the most fundamental and natural for modeling texts, it is useful to illustrate its framework.

(ii) It will be shown in a later section that our new classifier using the gamma-Poisson distribution clearly outperforms the original generative probabilistic classifier (multinomial naive Bayes) and is highly competitive with the support vector machine (SVM), which is the state of the art in terms of classification accuracy. This means that gamma-Poisson modeling is attractive not only for theoretical work but also for practical applications.

Note that the negative binomial distribution, which is the resultant posterior distribution of the gamma-Poisson pair [6], has been used for text modeling [5] and for text categorization [10]. In those studies, however, the hierarchical structure of gamma-Poisson modeling of texts was not taken into account, and the negative binomial distribution was merely used as a simple tool for describing word distributions. In the present work, the proposed classifier is based on hierarchical modeling; in other words, parameter estimation and classification procedures directly reflect the hierarchical structure of the gamma-Poisson distribution. In this sense, our approach is fundamentally different from the previous works.

To demonstrate that the proposed modeling is useful in practical applications, the classification accuracy and the computation time of the algorithm are examined using four standard datasets. The results lead us to conclude that the classifier with the proposed modeling is the most suitable among several tested classifiers in the case of noisy datasets. Furthermore, the advantage of the proposed model in incremental learning tasks that are frequently needed in practical application will be reported in detail.

The rest of the paper is organized as follows. Section 2 summarizes related works on the improvement of the generative probabilistic classifier in which some conjugate priors are utilized for better text modeling. In Section 3, we describe the framework of gamma-Poisson modeling of texts and how to construct a classifier by using the framework. Section 4 describes our experiments on automatic text classification. We summarize these results in Section 5 and discuss the characteristics of gamma-Poisson modeling in Section 6. In the last section, we give our conclusions.

2. Related Work

We first give a brief overview of the text categorization problem and introduce notation for later use. For the sake of completeness and later reference, we then review the original generative probabilistic classifier and its descendants, in which conjugate priors are used for better text modeling. Except for the beta-negative binomial model, all the models described in this section will be used in corresponding classifiers, and their performance will be compared with that of our new classifier through experiments.

2.1. Text Categorization Problem. Text classification/categorization is defined as the task of classifying documents into a fixed number of predefined categories. Categories are also called classes. Let $\mathscr{D}$ denote a domain of possible text documents, and let $C = \{c_1, c_2, \ldots, c_{|C|}\}$ be a finite set of predefined categories. In the conventional text categorization setting used in this study, each document $d \in \mathscr{D}$ is assigned to a single category $c \in C$. (Note that to simplify the problem, we consider here the categorization where each document is assigned to only one class. In other words, every document is assumed to be single labeled, not multilabeled. This assumption is made throughout this work.) We are given a set of training documents $D = \{d_1, d_2, \ldots, d_{|D|}\}$ which is a subset of $\mathscr{D}$, that is, $D \subset \mathscr{D}$. We assume that there is a target concept $\Phi : \mathscr{D} \to C$ that maps documents to categories. The result of the mapping $\Phi(d)$ is known for documents in the training set D; that is, each training document $d_i \in D$ has a class label $y_i \in \{c_1, c_2, \ldots, c_{|C|}\}$, which indicates that document d_i belongs to a category corresponding to the label. In the training phase, we attempt to find the classification function $f : \mathscr{D} \to C$, which approximates Φ from the information contained in training set D. In the test phase, f is used to classify new documents, the labels of which are unknown. The most important objective is to find f that maximizes accuracy (i.e., the percentage of times f and Φ agree).

To obtain a good f, the training set D must contain sample documents of all possible categories. If this is the case, D is expressed as

$$D = \bigcup_{c=c_1}^{c_{|C|}} D_c, \tag{1}$$

where D_c denotes a set of training documents that belong to a class c.

2.2. The Generative Probabilistic Classifier. The generative probabilistic classifier, often referred to as the *multinomial classifier* or *multinomial naive Bayes*, is one of the most popular classifiers for text categorization because it sometimes achieves good performance in various tasks, and because it is simple enough to be practically implemented even with a great number of features [11, 12]. The simplicity is mainly due to the following two assumptions. First, an individual document is assumed to be represented as a vector of word counts (bag-of-words representation). Since this representation greatly simplifies further processing, all the descendants including our new classifier inherit this first assumption. Next, documents are assumed to be generated by repeatedly drawing words from a fixed multinomial distribution for a given class, and word emissions are thus independent.

From the first assumption, documents can be represented as vectors of count-valued random variables. The ith document in a considered class c is then expressed as

$$d_{ci} = \left(x_{i1}, x_{i2}, \ldots, x_{ij}, \ldots, x_{i|V|}\right), \tag{2}$$

where x_{ij} is the count of the jth term t_j in the ith document in D_c and $|V|$ is a vocabulary size; in other words, we have

assumed here that the vocabulary of the considered dataset is given as $V = \{t_1, t_2, \ldots, t_{|V|}\}$ where t_j is the jth word in the vocabulary. From the second assumption, the probability of the document d_{ci} given by vector (2) is

$$p(d_{ci} \mid \theta_c) = \frac{\left(\sum_{j=1}^{|V|} x_{ij}\right)!}{\prod_{j=1}^{|V|}\left(x_{ij}!\right)} \prod_{j=1}^{|V|} \theta_{cj}^{x_{ij}}, \tag{3}$$

where θ_{cj} is the probability for the emission of t_j and is subject to the constraints $\sum_{j=1}^{|V|} \theta_{cj} = 1$. Note that for text classification, the parameters θ_{cj} must be evaluated for each possible class c. We use the estimator for θ_{cj} given by

$$\widehat{\theta}_{cj} = \frac{1 + \sum_{i=1}^{|D_c|} x_{ij}}{|V| + \sum_{i=1}^{|D_c|} \sum_{j=1}^{|V|} x_{ij}}, \tag{4}$$

where $|D_c|$ is the number of training documents belonging to the considered class c. To classify a new document with a given feature vector $d = (x_1, x_2, \ldots, x_{|V|})$, the multinomial classifier calculates a class-specific probability for class c as

$$p(c \mid d) \propto p(c)\, p(d \mid \theta_c) = p(c)\, \frac{\left(\sum_{j=1}^{|V|} x_j\right)!}{\prod_{j=1}^{|V|}\left(x_j!\right)} \prod_{j=1}^{|V|} \theta_{cj}^{x_j}, \tag{5}$$

where $p(c)$ is the prior probability of class c which is estimated from a training set by $p(c) = |D_c|/|D|$. We estimate θ_{cj} in (5) by using (4) for each specified class c. The document is assigned to the class with the highest probability $p(c \mid d)$.

The framework of the multinomial classifier described above usually works well in practical text classification tasks [11]. It has been pointed out, however, that the multinomial distribution used in the multinomial naive Bayes cannot describe the word burstiness phenomena that are inherently encountered in natural language documents [7]. Here, *burstiness* means the tendency of words in a document to appear in bursts; that is, if a word appears once, it is more likely to appear again. Since multinomial modeling is based on the assumption of independent word emissions with a fixed multinomial distribution for a considered class, the resultant word distribution fails to capture the burstiness especially for the words with moderate and low frequencies [7].

2.3. Dirichlet-Multinomial Model. A substantial improvement to describe the word burstiness phenomena has been achieved by introducing the Dirichlet-multinomial model, which models texts in a hierarchical manner to capture the burstiness [7]. In the model, the word count vector representing each document is generated by a multinomial distribution whose parameters are generated by its conjugate prior, that is, the Dirichlet distribution. The Dirichlet distribution is defined as

$$p(\theta_c \mid \alpha_c) = \frac{\Gamma\left(\sum_{j=1}^{|V|} \alpha_{cj}\right)}{\prod_{j=1}^{|V|} \Gamma\left(\alpha_{cj}\right)} \prod_{j=1}^{|V|} \theta_{cj}^{\alpha_{cj}-1}, \tag{6}$$

where $\alpha_c = (\alpha_{c1}, \alpha_{c2}, \ldots, \alpha_{c|V|})$ is a parameter vector of the Dirichlet of which components determine the density of the θ_c vector and where $\Gamma(\cdot)$ is the gamma function. The likelihood of a document $d = (x_1, x_2, \ldots, x_{|V|})$ with length n ($\sum_{j=1}^{|V|} x_j = n$) is given as an integral over θ_c vectors weighted by a Dirichlet distribution:

$$\begin{aligned} p(d \mid \alpha_c) &= \int_{\theta_c} p(d \mid \theta_c)\, p(\theta_c \mid \alpha_c)\, d\theta_c \\ &= \frac{n!}{\prod_{j=1}^{|V|} x_j!} \frac{\Gamma\left(\sum_{j=1}^{|V|} \alpha_{cj}\right)}{\prod_{j=1}^{|V|} \Gamma\left(\alpha_{cj}\right)} \frac{\prod_{j=1}^{|V|} \Gamma\left(x_j + \alpha_{cj}\right)}{\Gamma\left(\sum_{j=1}^{|V|} x_j + \alpha_{cj}\right)}, \end{aligned} \tag{7}$$

where we use the multinomial distribution function, (3), for $p(d \mid \theta_c)$. The classifier using Dirichlet-multinomial modeling computes class-specific probability of a given document d as

$$\begin{aligned} p(c \mid d) &\propto p(c)\, p(d \mid \alpha_c) \\ &\propto p(c) \frac{\Gamma\left(\sum_{j=1}^{|V|} \alpha_{cj}\right)}{\prod_{j=1}^{|V|} \Gamma\left(\alpha_{cj}\right)} \frac{\prod_{j=1}^{|V|} \Gamma\left(x_j + \alpha_{cj}\right)}{\Gamma\left(\sum_{j=1}^{|V|} x_j + \alpha_{cj}\right)} \end{aligned} \tag{8}$$

for each class c and assigns the document to the class with the highest probability. In the second expression of (8), we drop the term $n!/(\prod_{j=1}^{|V|} x_j!)$, which does not depend on class c. As in the case of θ_{cj} in multinomial modeling, one set of parameters for the Dirichlet, $\alpha_{cj} (j = 1 \cdots |V|)$, must be evaluated for each possible class. For the evaluation, we use the leave-one-out likelihood maximization method proposed by Minka [13] which offers a convergent fixed-point iteration for the update as

$$\alpha_{cj}^{\text{new}} = \alpha_{cj} \frac{\sum_{i=1}^{|D_c|} \left(x_{ij} / \left(x_{ij} + \alpha_{cj} - 1\right)\right)}{\sum_{i=1}^{|D_c|} \left(\sum_{j=1}^{|V|} x_{ij} / \left(\sum_{j=1}^{|V|} x_{ij} + \sum_{j=1}^{\nu} \alpha_{cj} - 1\right)\right)}. \tag{9}$$

It has been confirmed that the Dirichlet-multinomial leads to better text modeling in the sense that it can describe the burstiness for all word types ranging from frequent words to rare words. This success has recently led to two major modeling approaches along the lines of hierarchical modeling of texts: beta-binomial modeling and beta-negative binomial modeling.

2.4. Beta-Binomial Model. The beta-binomial distribution model is derived with consideration of a serious drawback in Dirichlet-multinomial modeling [9]. If we use the Dirichlet distribution, (6), to describe the probability density of θ_{cj}, then it is concluded that words having the same expectation value in θ_{cj} also have the same variance in θ_{cj}; that is, if $E[\theta_{cl}] = E[\theta_{cm}]$ then $\text{Var}[\theta_{cl}] = \text{Var}[\theta_{cm}]$. This is an undesirable property of the Dirichlet for text modeling because we aim to model different words as having the same expected value but different variances in order to describe various word occurrence patterns.

Allison [9] addressed this problem with the following assumptions.

(i) The probability of the occurrence of a document d is a product of independent terms, each of which

represents the probability of the number of emissions (i.e., the count) of an individual word.

(ii) The probability of the number of emissions is the average of a product $p(x_j \mid \theta_j)p(\theta_j \mid \alpha_j, \beta_j)$ over θ_j where θ_j is the probability for the emission of jth word t_j, $p(x_j \mid \theta_j)$ is the binomial distribution representing the probability of x_j times occurrence of t_j, and $p(\theta_j \mid \alpha_j, \beta_j)$ is beta-distributed weighting function of θ_j.

The assumptions described above allow means and variances for each θ_j to be specified separately and lead us to an expression of the probability of document d as

$$\begin{aligned} p(d \mid \alpha_c, \beta_c) &= \prod_{j=1}^{|V|} p\left(x_j \mid \alpha_{cj}, \beta_{cj}\right) \\ &= \prod_{j=1}^{|V|} \int_{\theta_{cj}} p\left(x_j \mid \theta_{cj}\right) p\left(\theta_{cj} \mid \alpha_{cj}, \beta_{cj}\right) d\theta_{cj} \\ &= \prod_{j=1}^{|V|} \binom{n}{x_j} \frac{B\left(x_j + \alpha_{cj}, n - x_j + \beta_{cj}\right)}{B\left(\alpha_{cj}, \beta_{cj}\right)}, \end{aligned} \tag{10}$$

where $B(\cdot)$ is the beta function and α_{cj} and β_{cj} are the parameters of the beta distribution. Note that the parameters α_j and β_j in the second assumption are replaced with α_{cj} and β_{cj} in the above equation since we aim to calculate $p(d \mid \alpha_c, \beta_c)$ for each class c to build a classifier. The classifier computes the class-specific probability $p(c \mid d) \propto p(c)p(d \mid \alpha_c, \beta_c)$ and the document d is assigned to the most probable class. Following Allison [9], we use the method of moments to evaluate sets of parameters α_{cj} and β_{cj} for each class.

2.5. Beta-Negative Binomial Model. The beta-negative binomial model is derived from consideration of the adequate empirical fit of the negative binomial distribution for text modeling [8]. The classifier using the model has been proved to be comparable to the multinomial naive Bayes in terms of classification performance, and this has been confirmed experimentally [8]. Hence, we do not test this model in the present experiment.

3. Gamma-Poisson Modeling of Text

3.1. Framework of the Model. As stated above, the Poisson distribution expresses the probability of a number of events occurring in a fixed period of time or in a specified unit of space. In text modeling, the "number of events" corresponds to the number of occurrences of a considered word in each document, all of which have a specified length in word count. Of course, documents are different from one another in length, and thus normalization, which will be described later, is necessary. A further step of text modeling is possible by extending the simple Poisson to a hierarchical gamma-Poisson description. The gamma-Poisson distribution is a Poisson distribution whose probability of the mean parameter λ follows a gamma distribution with a shape parameter α and a rate parameter β. The mass function of the Poisson describing the probability of x_j occurrences of the jth word is

$$p\left(x_j \mid \lambda_j\right) = \frac{\lambda_j^{x^j} \exp\left(-\lambda_j\right)}{x_j!}. \tag{11}$$

Its conjugate prior determining the density of λ_j is given by the gamma distribution, the density function of which is

$$p\left(\lambda_j \mid \alpha_j, \beta_j\right) = \frac{\beta_j^{\alpha_j}}{\Gamma\left(\alpha_j\right)} \lambda_j^{\alpha_j - 1} \exp\left(-\beta_j \lambda_j\right). \tag{12}$$

We incorporate the gamma-Poisson description in our model under assumptions that are similar to the beta-binomial case.

(i) The probability of document d can be decomposed to a product of independent terms, each of which represents the probability of a number of emissions for each individual word.

(ii) Each term can be expressed as the average of a product $p(x_j \mid \lambda_j)p(\lambda_j \mid \alpha_j, \beta_j)$ over λ_j where $p(x_j \mid \lambda_j)$ is the Poisson, (11), and $p(\lambda_j \mid \alpha_j, \beta_j)$ is gamma distributed, (12).

Consequently, the probability of document d is

$$\begin{aligned} &p(d \mid \alpha_c, \beta_c) \\ &= \prod_{j=1}^{|V|} p\left(x_j \mid \alpha_{cj}, \beta_{cj}\right) \\ &= \prod_{j=1}^{|V|} \int_{\lambda_{cj}} p\left(x_j \mid \lambda_{cj}\right) p\left(\lambda_{cj} \mid \alpha_{cj}, \beta_{cj}\right) d\lambda_{cj} \\ &= \prod_{j=1}^{|V|} \left\{ \frac{\Gamma\left(\alpha_{cj} + x_j\right)}{x_j! \, \Gamma\left(\alpha_{cj}\right)} \left(\frac{\beta_{cj}}{\beta_{cj} + 1}\right)^{\alpha_{cj}} \left(\frac{1}{\beta_{cj} + 1}\right)^{x_j} \right\}, \end{aligned} \tag{13}$$

where the parameters are replaced with class-specific ones. In the last expression of (13), we can see that each term of the products becomes a mass function of the negative binomial distribution when α_{cj} is an integer value [6]. The classifier computes the class-specific probability $p(c \mid d) \propto p(c)p(d \mid \alpha_c, \beta_c)$, and the document d is assigned to the most probable class.

3.2. Normalization of Document Length. To satisfy the conditions for using the gamma-Poisson description, documents must be normalized in length. Although several methods for normalizing document vectors have been proposed [12, 14, 15], we choose the simplest one that normalize a resulting vector in terms of L_1. The conversion of the jth component in a count-valued document vector is expressed as

$$x_j^{\text{new}} = \frac{x_j N}{\sum_{j=1}^{v} x_j}, \tag{14}$$

which gives the predefined fixed length of the normalized document as N. The normalization factor in the L_1 sense is $1/\sum_{j=1}^{v} x_j$ as seen in (14); in our experience, this factor leads to better classification performance than the common L_2 sense normalization with a factor $1/\sqrt{\sum_{j=1}^{v} x_j^2}$. This is because, in generative modeling of text, the document length of L_1 sense (i.e., total word count of a document) is considered to be the number of trials in which the selection of a considered word corresponds to a success. We set $N = 100$ for all documents because it allows intuitive understanding of the composition of normalized count vectors.

The normalization using (14) with $N = 100$ converts an integer-valued document vector into a real-valued one. This means that x_j in (13) changes from integer to real but it is not necessary to take into account the effects of this change when (13) is used in the classifier; the factorial $x_j!$ in (13) does not depend on class c, and we can safely omit $x_j!$ from the calculation of class-specific probability $p(c \mid d)$. Thus, the real-valued x_j does not cause any further difficulties in the evaluation of (13).

3.3. Estimation of Parameters. To compute the class-specific probability $p(c \mid d) \propto p(c)p(d \mid \alpha_c, \beta_c)$, we must use sets of parameters α_{cj} and β_{cj} $(j = 1 \cdots |V|)$ which are estimated for each specified class c from D_c (the set of training documents belonging to the class c). We use two different methods for the estimation: a rational approximation [16] and an iterative method [17].

The rational approximation estimates the parameters α_{cj} and β_{cj} through a set of equations:

$$\overline{x}_j = \frac{1}{|D_c|} \sum_{i=1}^{|D_c|} x_{ij}, \tag{15}$$

$$\widetilde{x}_j = \left(\prod_{i=1}^{|D_c|} x_{ij} \right)^{1/|D_c|}, \tag{16}$$

$$M_j = \ln \left(\frac{\overline{x}_j}{\widetilde{x}_j} \right), \tag{17}$$

$$\widehat{\alpha}_{cj} = \begin{cases} \dfrac{0.5000876 + 0.1648852M_j - 0.0544274M_j^2}{M_j} & \left(0 < M_j \leq 0.5772\right) \\ \dfrac{8.898919 + 9.059950M_j + 0.9775373M_j^2}{M_j\left(17.79728 + 11.968477M_j + M_j^2\right)} & \left(0.5772 < M_j \leq 17\right) \\ \dfrac{1}{M_j} & \left(17 < M_j\right), \end{cases} \tag{18}$$

$$\widehat{\beta}_{cj} = \frac{\widehat{\alpha}_{cj}}{\overline{x}_j}, \tag{19}$$

where x_{ij} is the count of jth word in ith document and $|D_c|$ is the number of documents belonging to the considered class.

The iterative method provides an update formula for α_{cj} as

$$\frac{1}{\alpha_{cj}^{\text{new}}} = \frac{1}{\alpha_{cj}} + \frac{\ln \widetilde{x}_j - \ln \overline{x}_j + \ln \alpha_{cj} - \Psi\left(\alpha_{cj}\right)}{\alpha_{cj}^2\left(1/\alpha_{cj} - \Psi'\left(\alpha_{cj}\right)\right)}, \tag{20}$$

where $\Psi(\cdot)$ and $\Psi'(\cdot)$ are the digamma and trigamma functions, respectively, and $\overline{x}_j$ and $\widetilde{x}_j$ are the same as in the rational approximation. The estimator for β_{cj} is still defined as $\widehat{\beta}_{cj} = \widehat{\alpha}_{cj}/\overline{x}_j$. In the iteration, we use

$$\alpha_{cj0} = \frac{0.5}{\ln \overline{x}_j - \ln \widetilde{x}_j} \tag{21}$$

as an initial value of α_{cj} [17] and apply a convergence criterion $|\alpha_{cj}^{\text{new}} - \alpha_{cj}| \leq 1.0 \times 10^{-5}$.

The resultant classification performance of each method in estimating parameters will be compared in the experimental section.

4. Experimental

To investigate the behavior of the proposed gamma-Poisson model described in the previous section, we perform experiments on automatic text categorization. In the experiments, the performance of a classifier using gamma-Poisson modeling is compared with the performance of other classifiers that also use probabilistic modeling but with different distributions. As mentioned above, the models selected for the comparison are the multinomial, the Dirichlet-multinomial, and the beta-binomial. In our experiments, the support vector machine (SVM) is also used as a standard discriminative classifier because previous comparative studies on classifiers [14, 18] have consistently shown that SVM is the state of the art in terms of classification accuracy.

4.1. Data Set. For our experiments, we use four different datasets that are chosen to represent a wide spectrum of text classification tasks.

The first one is the 20 Newsgroups dataset which was originally collected with a netnews-filtering system [19] and contains approximately 20,000 documents being partitioned (nearly) evenly across 20 different UseNet newsgroups. We use the 20news-18828 version (original data set is available from http://kdd.ics.uci.edu/databases/20newsgroups/20newsgroups.data.html. 20News-18828 is available from http://people.csail.mit.edu/jrennie/20Newsgroups/) from which cross-posts have been removed to give a total of 18,828 documents. Consequently, 20 Newsgroups is a single labeled dataset with approximately even class distribution, and the task is to apply one of the 20 possible labels to each test document. We build an initial vocabulary from all words left after stop word, punctuation, and number token removals. Capital letters are transformed to lowercase letters and no stemming algorithm is applied. Here, words are defined as alphabetical strings enclosed by whitespace. The size of the initial vocabulary is 103,135 words.

The second dataset is the Reuters-21578 data collection (data set is available from: http://kdd.ics.uci.edu/databases/reuters21578/), which contains documents that appeared on the Reuters newswire in 1987 and were manually classified by personnel from Reuters Ltd. For applications in which enough numbers of training and test documents are required, the top 10 categories (the 10 categories with the highest number of positive training examples in the ModApte split) are usually used [2]. However, since we want to use a single labeled dataset as stated in Section 2.1, we make a slight modification to the usual top 10 categories; specifically, we eliminate all documents with more than one topic (category); in this way, two categories among the top 10 are excluded. We use documents in the resultant 8 categories as our dataset. Consequently, the task is applying one of the 8 possible labels to each of test documents. In contrast to the 20 Newsgroups, the class distribution of this dataset is quite imbalanced; the largest topic category "earn" has 3,923 documents while the smallest "grain" has only 51 documents. The preprocessing used to build an initial vocabulary is the same as for the 20 Newsgroups, and the resulting vocabulary has 22,793 words.

The third test collection is the Industry Sector dataset which is a collection of corporate web pages organized into hierarchical categories based on what a company produces or does. Although it has a hierarchy with three levels of depth, we do not take the hierarchy into account and use a flattened version of the dataset. This dataset contains a total of 9,555 documents divided into 104 categories. (We obtained the dataset from http://www.cs.umass.edu/mccallum/code-data.html. Because it was found that one of the original 105 categories was empty, the remaining 104 categories having documents were used in our experiments.) We use all 9,555 documents in our experiments without removing the multilabeled documents because the fraction of multilabeled documents is very small and the effect of these documents is negligible (only 15 documents out of 9,555 belong to two classes; thus, they cannot affect our results considerably). The largest and smallest categories have 105 and 27 documents, respectively, and the average number of documents per category is 91.9. For this dataset, we remove HTML tags by skipping all characters between "<" and ">", and we did not use a stop list. The resulting vocabulary has 64,680 words.

The fourth test collection is the TechTC-100 dataset which is a collection of web pages taken from the web directory of the Open Directory Project (ODP) (the TechTC-100 dataset is available from http://techtc.cs.technion.ac.il/techtc100/). Because this test collection was generated from the web directory in a fully automated manner [20], it is noisier than the other three test collections; for example, textual advertisements included in the web pages can be noise that affects the classification accuracy. The original TechTC-100 dataset contains 100 datasets, each of which consists of positive and negative documents that define a binary classification task, and these positive and negative documents are chosen from the pairs of the ODP categories. Although there are 100 different combinations of positive and negative categories in the datasets, we only use positive documents because positive documents for each class are sufficient to define a multiclass classification problem. We found that 40 distinct positive categories in the 100 pairs of positive and negative categories, and therefore the task becomes applying one of the 40 possible labels to each of test documents. We did not apply the preprocessing steps to this test collection because it is supplied in a preprocessed plain text format. This collection has a vocabulary of 103,003 words.

For all four datasets, we use 10-fold cross-validation to make maximal use of the data and to allow comparison with the previous work by Allison [9]. Ten obtained values of performance are averaged to give the final result.

4.2. Vector Creation. To investigate the effect of vocabulary size on classification performance, we use a simple feature selection method based on the collection term frequency as follows. First, we count the collection term frequency, CF, which is the total frequency of each word throughout the entire dataset. Second, we select all words that satisfy CF $\geq$ N_0 where N_0 is a predefined integer. The feature selection by CF is one of the simplest methods, but is sufficient for the task at hand, namely, comparing different classifiers at each vocabulary size. The resultant vocabulary sizes after feature selection are summarized in Table 1.

Two different types of document vectors, namely, count-valued and normalized, are used to represent each document. A count-valued document vector is constructed from document term frequency (number of occurrences of a considered word in a document) for each word, and then each component in the vector is converted by use of (14) to give the normalized vector. The count-valued document vectors are supplied to the classifiers with multinomial, Dirichlet-multinomial, and beta-binomial modeling as training and test data. The normalized document vectors are supplied to the classifier with gamma-Poisson modeling that inherently requires the normalization. For SVM, we use both types of document vectors and find that the normalized vectors give better classification performance. We will thus show the performance of SVM with only normalized vectors.

4.3. Algorithm and Complexity. The algorithm of our classifier using gamma-Poisson modeling is shown in Algorithm 1 for the training and test phases. In the training phase, the classifier estimates the values of parameters $\widehat{\alpha}_{cj}$ and $\widehat{\beta}_{cj}$ for each class, from given training vectors; in the test phase, the classifier assigns the most probable label to a given test vector. As seen from the pseudocode in Algorithm 1, the time complexity of our classifier at the training phase with rational approximation of parameters is $O(|D||V|)$ where D and V denote the set of all training documents and the vocabulary (the set of all terms satisfying CF $\geq$ N_0 in a corpus), while the complexity at the test phase is given by $O(|C||V|)$. The classifiers using multinomial and beta-binomial modeling also have the same complexity, $O(|D||V|)$ and $O(|C||V|)$, for the training and test phases, respectively, because they basically have the same code structure in our implementations and the differences are only in the equations for calculating parameters and estimating class-specific probabilities.

Table 1: Vocabulary size obtained by feature selection with CF.

Feature selection	20 Newsgroups	Reuters-21578	Industry Sector	TechTC-100
Initial vocabulary	103,135	22,792	64,680	103,003
CF ≥ 2	63,285	13,809	38,107	55,250
CF ≥ 5	34,152	7,268	21,681	29,070
CF ≥ 10	21,845	4,455	14,767	18,946
CF ≥ 20	13,792	2,626	9,885	12,254
CF ≥ 50	7,166	1,224	5,722	6,593
CF ≥ 100	4,056	689	3,580	3,937
CF ≥ 200	2,091	345	2,057	2,249
CF ≥ 500	693	110	854	936
CF ≥ 1000	230	36	420	432
CF ≥ 2000	57	18	159	195

gamma-Poisson classifier for training phase

01 **for** each $c \in C$
02 **for** each $t \in V$
03 **for** each $d_c \in D_c$
04 $x_t \leftarrow$ count tokens of term t in d_c
05 $\overline{x}_t \leftarrow \overline{x}_t + x_t$
06 $\widetilde{x}_t \leftarrow \widetilde{x}_t + \log(x_t)$
07 **end for**(d_c)
08 $\overline{x}_t \leftarrow \overline{x}_t / |D_c|$
09 $\widetilde{x}_t \leftarrow \widetilde{x}_t / |D_c|$
10 $M_t \leftarrow \log(\overline{x}_t) - \widetilde{x}_t$
11 $\alpha_{c,t} \leftarrow$ rational Approximation(M_t)
12 $\beta_{c,t} \leftarrow \alpha_{c,t} / \widetilde{x}_t$
13 **end for**(t)
14 **end for**(c)

gamma-Poisson classifier for test phase

01 **for** each $c \in C$
02 score$[c] \leftarrow \log p(c) + \log p(d \mid \alpha_c, \beta_c)$
03 **end for**(c)
04 **return** argmax$_{c \in C}$ score$[c]$

Algorithm 1: Algorithm of classifier using gamma-Poisson modeling. In the training phase, the procedure of learning over D (entire training documents) is given, while in the test phase, the procedure to classify one test document d is described. D_c is the set of training documents belonging to a class c and d_c is a document vector in D_c. Note that $\sum_c |D_c| = |D|$, and thus the time complexity of the training phase is estimated as $O(|D|\,|V|)$. For the test phase, the complexity is found to be $O(|C|\,|V|)$ because $\log p(d \mid \alpha_c, \beta_c)$ in line 02 of the pseudo code is calculated through the entire summation over $|V|$ (see (13)).

On the other hand, the classifier using Dirichlet-multinomial modeling and using gamma-Poisson modeling *with iterative approximation* ((20)) has complexity given by $O(|D||V|n_{\mathrm{it}})$ at the training phase, where n_{it} is the number of iteration cycles required for the convergence of the parameters. This is usually larger than $O(|D||V|)$ since n_{it} is typically of the order of ten and Dirichlet-multinomial modeling is therefore expensive in terms of computation time, as will be confirmed later.

Note that the time complexities described here ($O(|D||V|)$ or $O(|D||V|n_{\mathrm{it}})$ for the training phase and $O(|C||V|)$ for the test phase) are all linearly dependent on the vocabulary size $|V|$. This linear dependence will be confirmed empirically through comparisons of the practical computation times in the next section.

4.4. Implementation Issues. Except SVM, all the classifiers are implemented in the Java programming language. Supplementary information is as follows.

(i) In the learning phase of the classifier utilizing the Dirichlet-multinomial model, initial values of α_{cj} in the iterative evaluation with (9) are set to $\alpha_{cj} = 0.5$ for all j. When an estimated value of α_{cj} is equal to zero, which occurs when the corresponding jth term failed to appear in all the training documents in

considered class D_c, we replace the value with $\alpha_{cj} = 1.0 \times 10^{-20}$. This smoothing is similar to the method used by Madsen et al. [7].

(ii) For the classifier with the beta-binomial model, the estimated α_{cj} also becomes zero when the corresponding jth term fails to appear at the estimation using the method of moments. As proposed by Allison [9], to prevent any α_{cj} from being zero, we supplement actual training documents with a pseudo-document in which every word occurs once.

(iii) For the classifier with the gamma-Poisson model, if x_{ij} in (15) and (16) is zero, then we replace the value with $x_{ij} = 0.001$ to prevent the geometrical means, $\tilde{x}_j$ defined by (16), from being zero (we preliminarily tried four values of x_{ij} for the replacement, namely, $x_{ij} = 0.1, 0.01, 0.001, 0.0001$, and obtained the best performance when $x_{ij} = 0.001$). Further, if the arithmetic mean given by (15) and the geometric mean given by (16) are equal, which happens when the corresponding jth term fails to appear for all i (in all the training documents), we set $M_j = 0.001$ in (18).

(iv) To compute several special functions, namely, the gamma function in (8) and (13), the beta function in (10), and the digamma and the trigamma functions in (20), components offered by the Apache Commons Mathematics Library (the library is available from http://commons.apache.org/proper/commons-math/) are used.

(v) For the SVM classifier, we use SVM$^{\text{multiclass}}$ which is one of the popular implementations (the implementation is available from http://svmlight.joachims.org/svm_multiclass.html). of the multi-class support vector machine In the training phase of SVM, the trade-off parameter between training error and margin, c, is set to 5,000 to obtain high accuracy. For all other parameters, we use default values.

(vi) The experiments are conducted on a PC with a Phenom II X4 3.4 GHz processor and 8 GB of RAM.

(vii) A pilot implementation of the gamma-Poisson classifier using the C programming language is about 2.5 times faster than that using the Java. One should bear in mind this difference when comparing absolute computation times of our classifier with those of SVM, because the SVM classifier is implemented in C and the computation times of our classifier shown in the next section are those using the Java implementation.

5. Results

In this study, we use the simplest measure of classification performance, that is, accuracy, which is simply defined as a ratio of the total number of correct decisions to the total number of test documents in the dataset used. Note that for a single labeled dataset and a single labeled classification scheme as in this work, the microaveraged precision and recall are equivalent and hence equal to the *F1* measure [23], which we termed here "accuracy".

Table 2: Classification accuracy on the 20 Newsgroups dataset with two different methods for estimating parameters for the gamma-Poisson distribution. Values are shown as accuracy $\pm\sigma$ where σ is the standard deviation calculated through 10-fold cross-validation.

Feature selection	Accuracy	
	Rational approximation	Iterative method
Initial vocabulary	0.9148 ± 0.0066	0.9146 ± 0.0062
CF ≥ 5	0.9132 ± 0.0057	0.9133 ± 0.0058
CF ≥ 10	0.9084 ± 0.0072	0.9081 ± 0.0073
CF ≥ 20	0.9002 ± 0.0068	0.9005 ± 0.0063
CF ≥ 50	0.8775 ± 0.0079	0.8773 ± 0.0081
CF ≥ 100	0.8533 ± 0.0095	0.8532 ± 0.0094

5.1. Parameter Estimation for Gamma-Poisson Modeling. We begin by considering the validity of the parameter estimation methods for gamma-Poisson modeling that were described in Section 3.3. As seen in Table 2, the accuracy values obtained for the two estimation methods are almost equivalent, indicating that the precision of the rational approximation and the convergence of the iterative method are both sufficient. From this result, it is confirmed that both methods are valid for estimating parameters, although in terms of computational speed, the rational approximation is superior to the iterative method. We will show the results only for the rational approximation in the next section.

5.2. Performance Comparison in Text Classification Tasks for 20 Newsgroups and Reuters-21578

5.2.1. Classification Accuracy. The main purposes of this work are to demonstrate the gamma-Poisson model as a new tool for text modeling and to show the extent to which it can be appropriately used in text classification tasks. In this sense, the performance comparison between the classifier using the gamma-Poisson model and those using other models are our primary result in this work. Figure 1 shows the performance comparison of various classifiers in the text classification task for the 20 Newsgroups dataset, and Figure 2 shows the same for the Reuters-21578 data collection. The exact vocabulary sizes at each data point in these figures are given in Table 1. In Figure 1, the best performance is seen for SVM, but the classifiers using the beta-binomial and gamma-Poisson models are almost equivalent and are highly competitive with SVM; they are inferior to SVM only in the range of limited vocabulary size below 20,000 words. The classifiers using the multinomial and the Dirichlet-multinomial models are apparently worse than the other three classifiers in terms of classification accuracy.

In Figure 2, SVM is again the best performer. An important difference between Figures 1 and 2 is that the classifier with the gamma-Poisson is superior to that with the beta-binomial in Figure 2, whereas they are almost equivalent

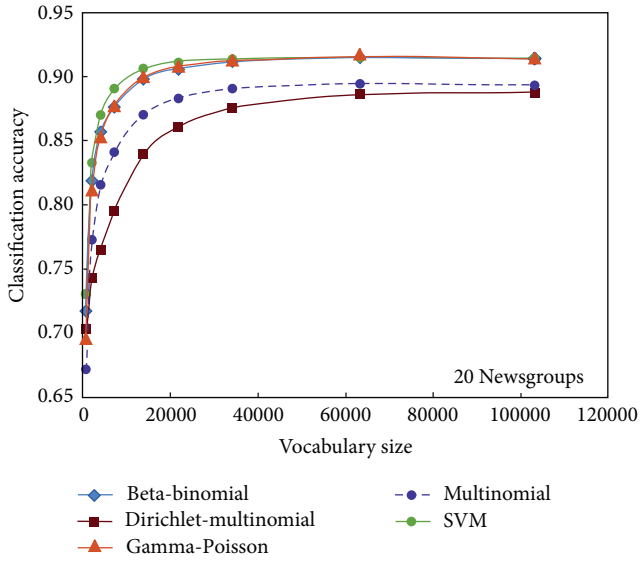


Figure 1: Performance of various classifiers for 20 Newsgroups dataset.

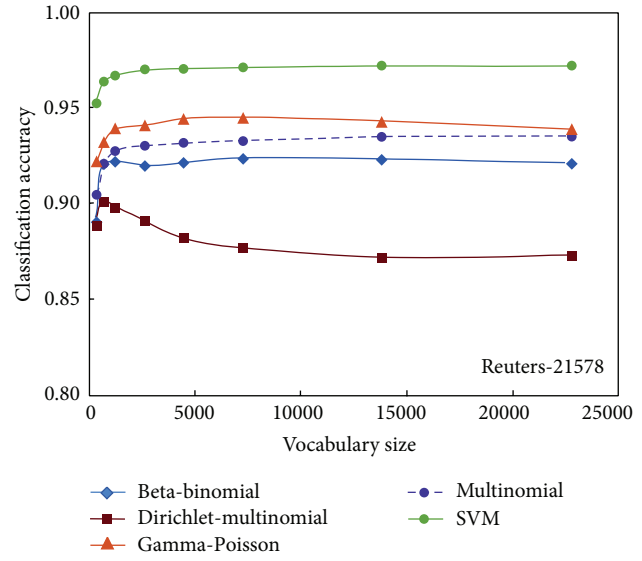


Figure 2: Performance of various classifiers for Reuters-21578 dataset.

in Figure 1. Another point is that the multinomial classifier performs better than the beta-binomial in Figure 2 but it was worse in Figure 1. The classifiers with the Dirichlet-multinomial exhibit the worst performance in both cases.

5.2.2. Computation Time. Figures 3 and 4, respectively, show comparisons of computation times for various classifiers in the tasks for the 20-Newsgroups and Reuters-21578 datasets. Note that the computation time of each classifier is defined here as the sum of the training time and the test time; the former is the time needed in order to estimate parameters from training vectors while the latter is the time needed to assign the most probable labels to test vectors. From the requirement of 10-fold cross-validation, the ratio of the numbers of training to test vectors is 9 : 1 and the sum of these gives the total number of documents in a considered corpus. Measured 10 values of computation time through 10-fold cross validation are averaged and used as a final result in these figures. In Figures 3(a) and 4(a) the horizontal and the vertical axes are represented in linear scales while they are shown logarithmically in Figures 3(b) and 4(b) in order to show the lower vocabulary region clearly.

In Figures 3 and 4 the computation times of all the classifiers except SVM show almost linear dependence on the vocabulary size which is consistent with the time complexities of these classifiers described in Section 4.3. Clearly, SVM is very fast except in the limited vocabulary region. The classifier using beta-binomial modeling is slower than that using the gamma-Poisson because the method of moments used in the training phase of the beta-binomial classifier is time consuming compared with the rational approximation used in the gamma-Poisson classifier. The Dirichlet-multinomial classifier is the slowest because of the iterative procedures for estimating parameters.

5.3. Performance Comparison in Text Classification Tasks for Industry Sector Dataset. Figures 5 and 6 show the classification accuracy and the computation time, respectively, in text classification for the Industry Sector dataset. As clearly seen in these figures, SVM is still the best performer in terms of the classification accuracy; however it is the worst in terms of the computation time. (We first used the binary version of SVM$^{\text{multiclass}}$ for Windows and found that the SVM$^{\text{multiclass}}$ suffers of memory errors when it is adapted to the Industry Sector and the TechTC-100 datasets. To avoid the error, we then used the SVM$^{\text{multiclass}}$ on Linux which was compiled with GCC. The results of SVM shown in this study were those obtained on Linux. The reason for the error is probably due to not enough heap memory available for the SVM$^{\text{multiclass}}$ on Windows.) The worst computation time of SVM indicates that this dataset has fundamentally different, undesirable characteristics for SVM. As will be discussed in the next section, the slow computation time of SVM is attributable to this dataset being not linearly separable because of its noisy nature. For the Industry Sector dataset, a reasonable and well-balanced choice of classier is found to be the gamma-Poisson because it achieves the second best accuracy and requires moderate computation time.

5.4. Performance Comparison in Text Classification Tasks for TechTC-100. Figures 7 and 8 show the results of classification accuracy and those of computation time, respectively, at the task on the TechTC-100 dataset. The results are very similar to those seen in the Industry Sector dataset; that is, SVM wins in terms of the classification accuracy but losses in terms of the computation time. Note that the computation time of gamma-Poisson, which gives the second best classification accuracy as in the case of the Industry Sector dataset, is about ten times faster than that of SVM.

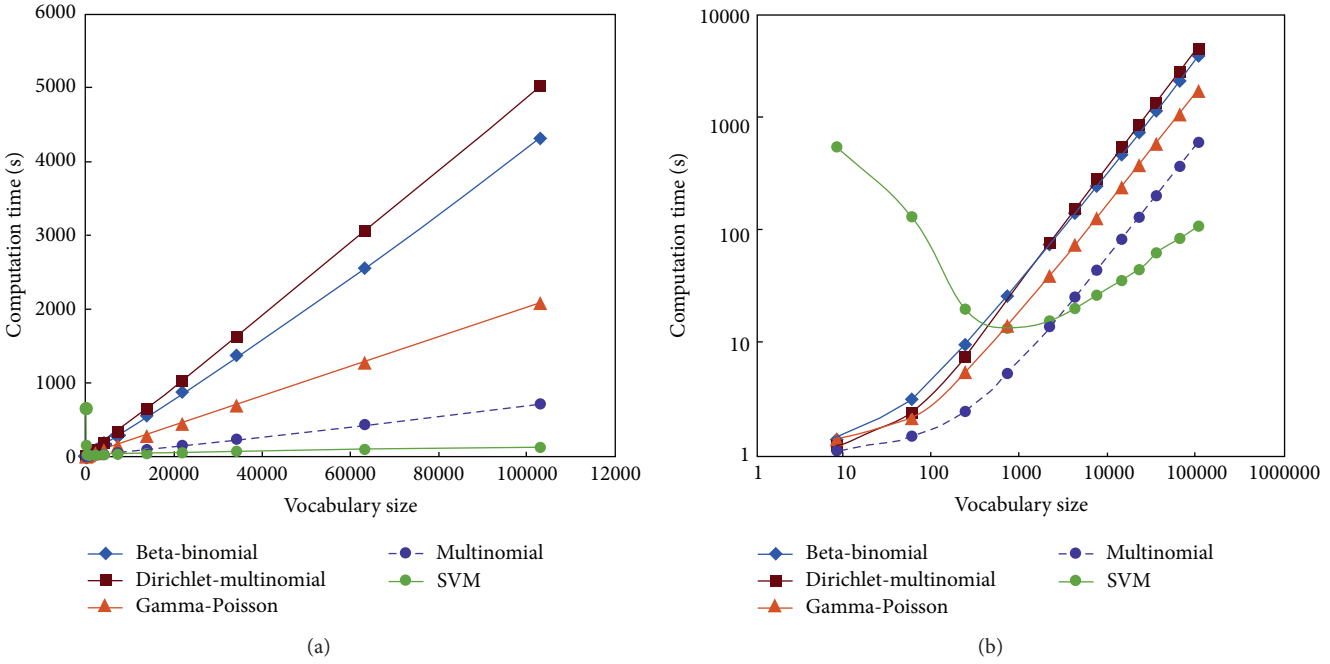


Figure 3: Computation time of various classifiers for 20 Newsgroups dataset.

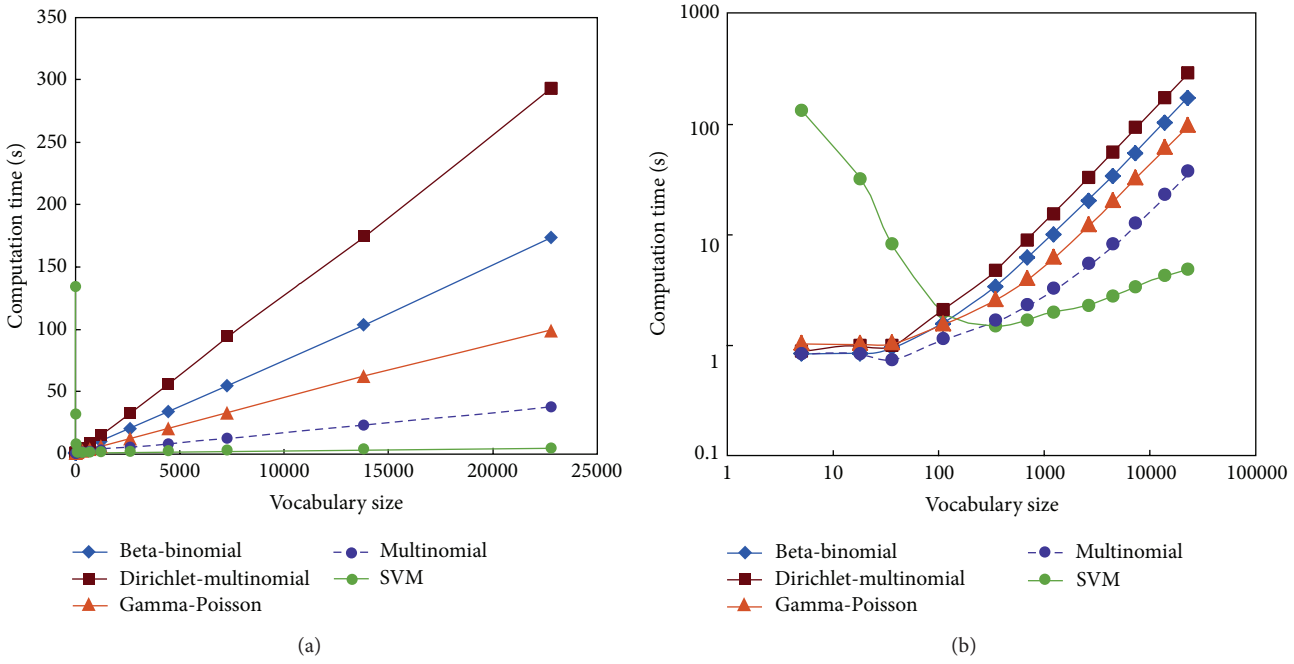


Figure 4: Computation time of various classifiers for Reuters-21578 dataset.

6. Discussion

6.1. Performance of the Probabilistic Classifiers. As seen in the previous section, the overall trend in classification performance for the five tested classifiers can be summarized as follows:

$$\text{SVM} \geq \text{gamma-Poisson} \geq \text{beta-binomial} \geq \text{Multinomial} > \text{Dirichlet-Multinomial}, \quad (22)$$

where the symbols ">" and "≥" should be read as "better than" and "better than or equivalent to", respectively. Among the results, we first consider why the gamma-Poisson and the beta-binomial are superior to the Dirichlet-multinomial. The point is that these three classifiers perform differently; nevertheless, they have similar structures in terms of hierarchical text modeling with utilizing conjugate priors. This result probably arises from a difference in the properties of the conjugate priors used. As mentioned earlier, we cannot

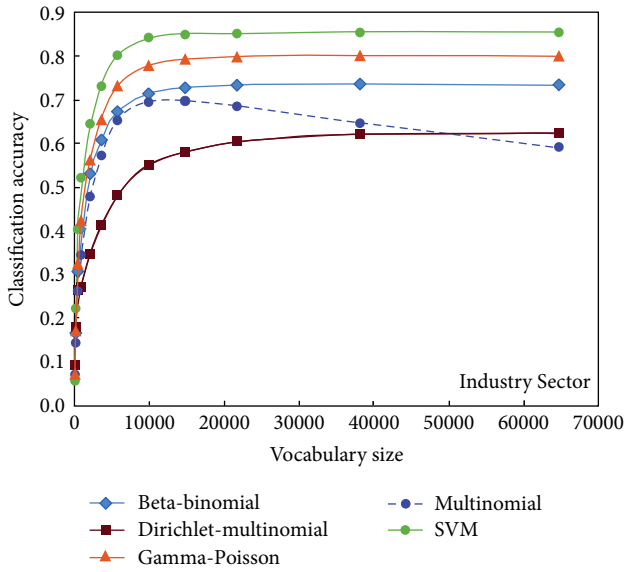


FIGURE 5: Classification performance of various classifiers for the Industry Sector dataset.

specify means and variances separately in the Dirichlet distribution while separate specification is possible for the beta distribution. Note that separate specification is also possible in the case of the gamma distribution; under the notations used in (12), the mean and variance of the gamma distribution are expressed as

$$E\left[\lambda_j\right] = \frac{\alpha_j}{\beta_j}, \qquad V\left[\lambda_j\right] = \frac{\alpha_j}{\beta_j^2}, \tag{23}$$

and hence

$$\alpha_j = \frac{\left(E\left[\lambda_j\right]\right)^2}{V\left[\lambda_j\right]}, \qquad \beta_j = \frac{E\left[\lambda_j\right]}{V\left[\lambda_j\right]}. \tag{24}$$

Equations (23) and (24) give us the separate specification which guarantees a more flexible description of word occurrence in gamma-Poisson modeling compared with the description in Dirichlet-multinomial modeling. Thus, the origin of superiority of the beta-binomial and the gamma-Poisson over the Dirichlet-multinomial can be attributed to their flexibility to describe various patterns of word distribution.

We next consider the reason that the gamma-Poisson gives somewhat better performance than the beta-binomial. As described in any textbook on probability distribution, the binomial can be approximated by the Poisson when the number of trials goes to infinity and the expected number of successes remains fixed. It is therefore reasonable to expect that the gamma-Poisson is almost equivalent to the beta-binomial but never exceeds it. Our expectation, however, is betrayed as seen in Figures 2, 5, and 7. A possible interpretation of this result is that the better performance of the gamma-Poisson model is attributable to the normalization of document vectors. In the beta-binomial model, each term occurrence is treated as being equally important in the estimation of parameters with the method of moments and in the classification of test documents. However, in vectors normalized in the L_1 sense, the event of a word occurrence has remarkably different weight according to the original document length, and in our experiments these normalized vectors are only used for the gamma-Poisson and the SVM classifiers as training and test vectors. In the vector normalization, the word occurrence in a short document is converted to be more heavily weighted than that in a long document. The conversion in this manner is reasonable and considered to bring about the better performance because short documents usually have fewer unnecessary terms that are irrelevant to the topic, and the ratio of informative terms that represent a concept of the topic is higher than in long documents. From this aspect of the vector normalization, further discussion of a condition for improving accuracy is possible. If each document has almost the same length, then the normalization does not change the weight of word occurrences and thus does not contribute to the improvement of accuracy. Therefore, the normalization of document vectors can be effective in the situation where the distribution of document length is scattered in a considered dataset.

To confirm this, we introduce a measure that quantifies how many terms are used in a document vector. The measure we tentatively use here is a ratio of nonzero components defined as

$$\begin{aligned}&(\text{ratio of nonzero components})\\ &= \frac{\#\text{ nonzero components in the document vector}}{\#\text{ all components comprising the document vector}}.\end{aligned} \tag{25}$$

For this ratio, we expect that when the distribution of the ratio becomes broader, the difference in accuracy between the gamma-Poisson and the beta-binomial will become increasingly evident. Figure 9 shows the distributions of the ratio of nonzero components for the four used datasets; the x-axis represents the ratio and the y-axis is the total number of documents which have that ratio of nonzero components. Statistic summaries of these four distributions are given in Table 3. In the table, we also give the values of

$$\Delta = \frac{P_{\mathrm{GP}} - P_{\mathrm{BB}}}{P_{\mathrm{BB}}}, \tag{26}$$

where P_{GP} is the classification accuracy of the gamma-Poisson classifier at the full vocabulary size and P_{BB} is that of the beta-binomial classifier. Since Δ represents the difference in performance in percent between the gamma-Poisson and the beta-binomial, we can test our hypothesis on the vector normalization described above by examining the correlation between Δ and other statistic measures. As confirmed from the values of interquartile range (IQR) and standard deviation (SD) in Table 3 and also as intuitively seen in Figure 9, 20 Newsgroups has the sharpest distribution, and it consistently has the smallest Δ. The ratio is more broadly distributed in the Reuters-21578 and TechTC-100 datasets, and Δ thus takes larger values. These results indicate that

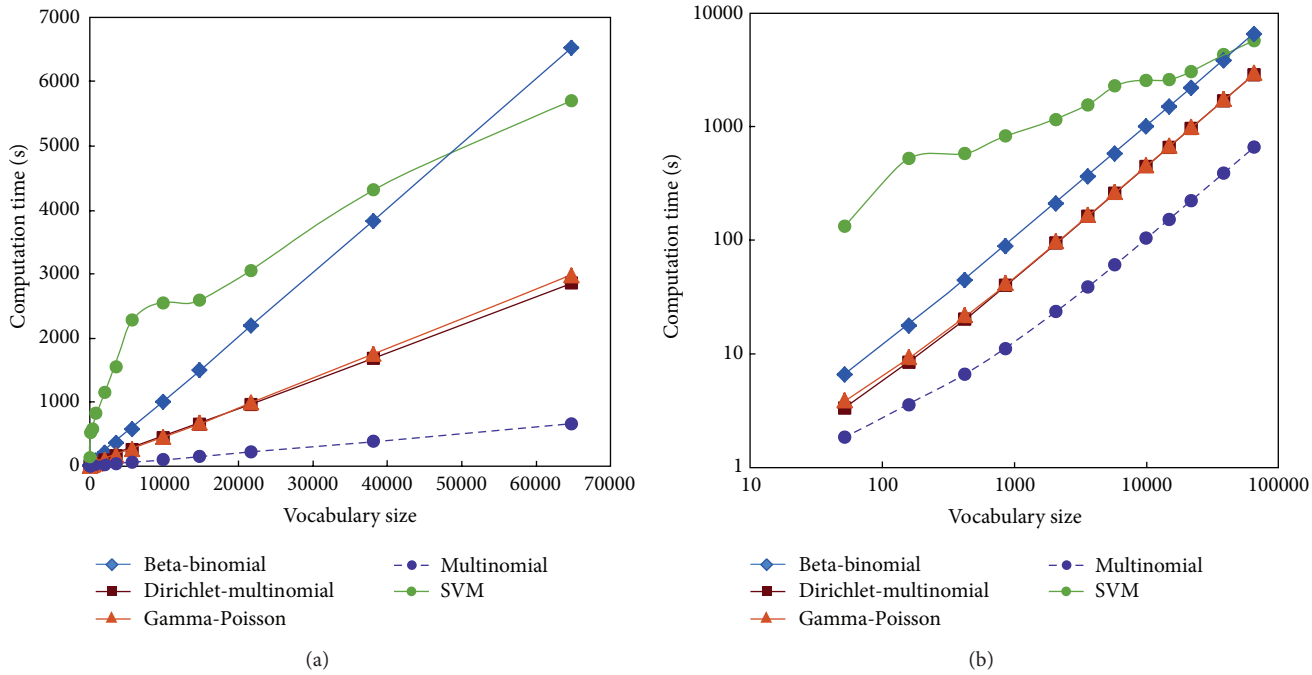


Figure 6: Computation time of text classification task for Industry Sector dataset.

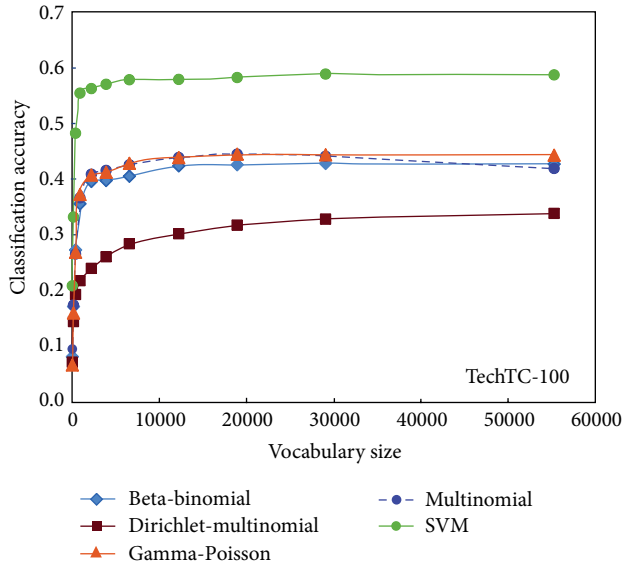


Figure 7: Performance of various classifiers for TechTC-100 dataset.

there is a positive correlation between Δ and the broadness of distribution, which supports our hypothesis regarding vector normalization. In comparison, the Industry Sector seems to have an unusually large value of Δ in relation to its IQR and SD values. However, this can be explained by the portion of very short documents being the largest in this dataset, which can be clearly seen in Figure 9. Since the improvement of accuracy with the vector normalization is more effective in short documents, the largest Δ for the Industry Sector dataset is consistent with our hypothesis. The superiority of the gamma-Poisson over the beta-binomial observed for the Reuters-21578, Industry Sector, and TechTC-100 datasets is therefore explained, at least partially, in terms of the effectiveness of the vector normalization.

Table 4 lists our best classification results for 20 Newsgroups using all the words in the initial vocabulary. In the table, data obtained by other authors are also shown for comparison.

As shown in the table, our results agree reasonably well with those by the other authors. A detailed comparison is given below.

(i) Our result for the Dirichlet-multinomial is similar to that reported by Madsen et al. [7], whereas the same model was found to perform worse in the study of Allison [9]. This disagreement is attributed to the difference in the estimation of parameters; the iterative methods used in [7] and in this work give, in general, a better estimation than the method of moments used by Allison [9].

(ii) Our result for the beta-binomial is similar to the result reported by Allison [9]. This is because we made efforts to ensure maximal comparability with the work of Allison [9] for fair comparison.

(iii) The difference among the three results for SVM probably arises from the difference of term weighting methods to create document vectors and from the difference in the parameter settings of SVM. For document vectors, we use normalized vectors in the L_1 sense while more sophisticated term frequency-inverse document frequency (TF-IDF) was used by Kibriya et al. [22]. All parameters were set to be their default values in the study by Allison [9], whereas we used a large value of c to obtain higher accuracy.

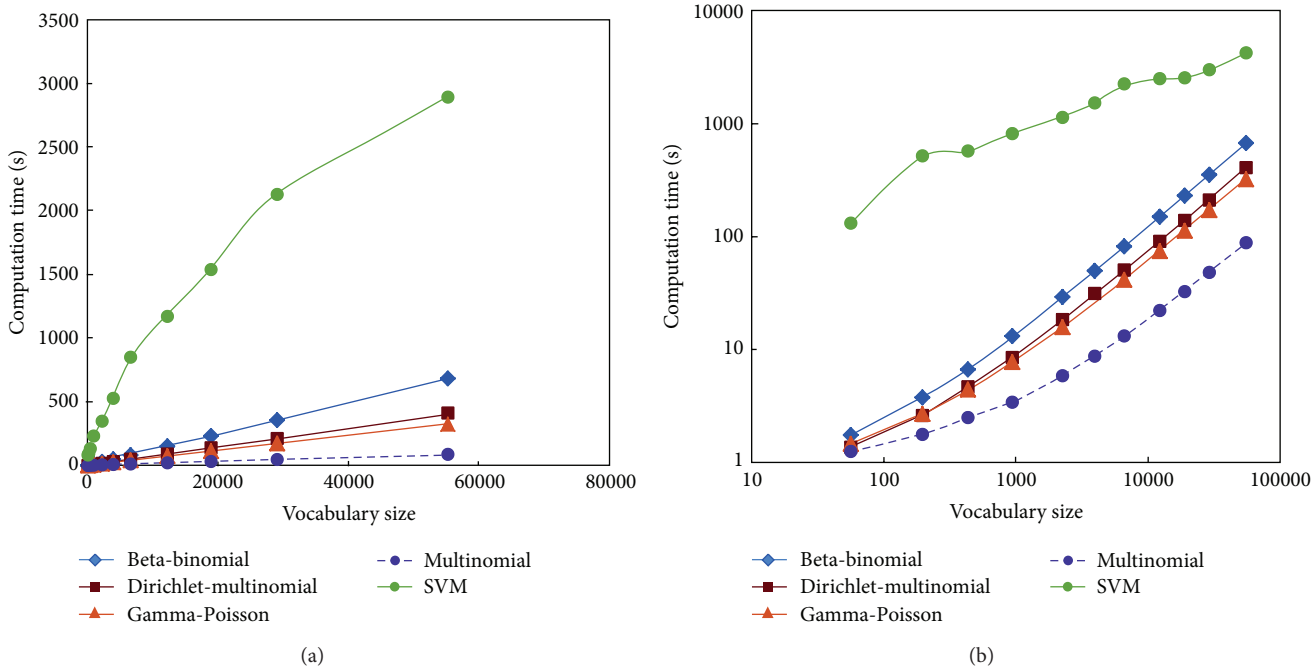


Figure 8: Computation time of text classification task for TechTC-100 dataset.

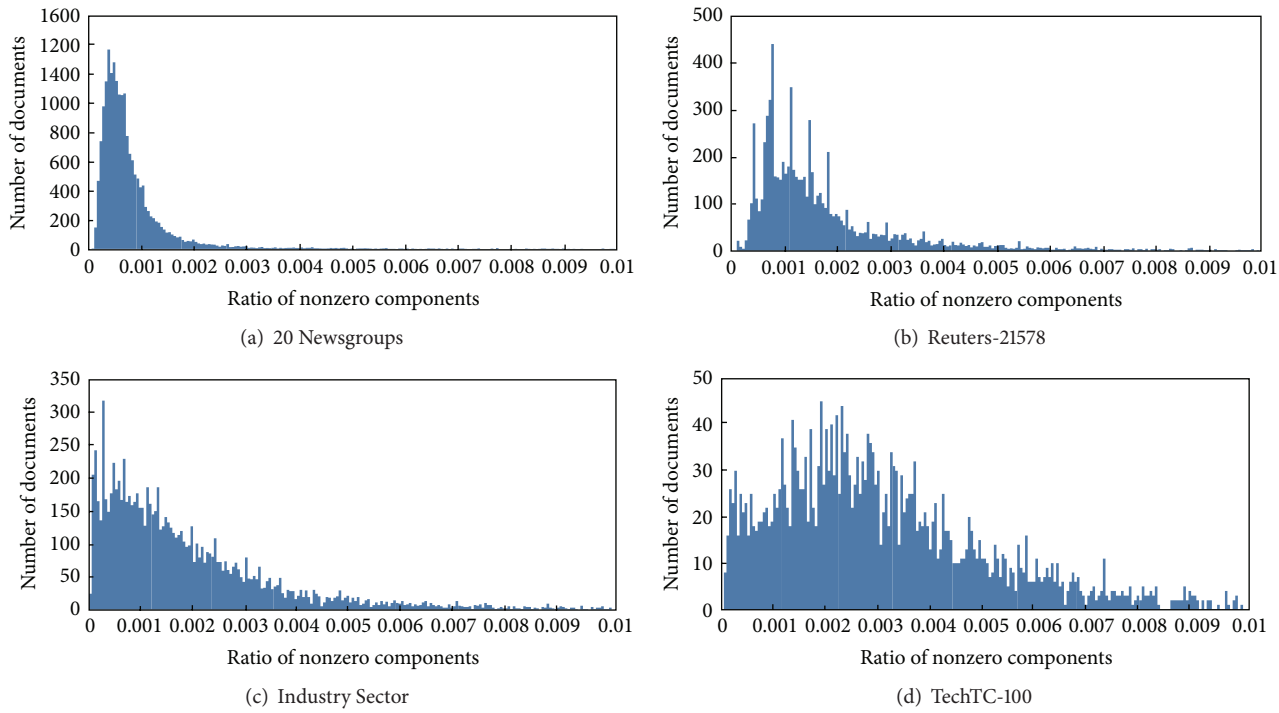


Figure 9: Distribution of the ratio of nonzero components for the four datasets used here. In all cases, t-he initial vocabularies were used to create document vectors.

(iv) The result of the multinomial in this work is almost identical to the result reported by Joachims [21], while Allison [9] and Madsen et al. [7] reported much worse values. The origin of this difference might be attributable to the differences in preprocessing (stop word removal and stemming) and vocabulary size. Although we obtained relatively high classification accuracy for the multinomial in this work, it is clear that the gamma-Poisson outperforms the multinomial by some margin.

Table 3: Statistical summary of the distribution of nonzero components for the four datasets used here. IQR and SD mean inter quartile range and standard deviation, respectively.

Dataset	Δ (%)	Median	Mean	IQR	SD
20 Newsgroups	0.05	0.00054	0.00077	0.0005	0.0011
Reuters-21578	1.97	0.00123	0.00167	0.0012	0.0015
Industry Sector	9.04	0.00135	0.00206	0.0019	0.0026
TechTC-100	3.70	0.00276	0.00352	0.0027	0.0037

6.2. Comparison of Gamma-Poisson Classifier and SVM. Although the proposed classifier using gamma-Poisson modeling fails to outperform the SVM classifier as seen in the previous section, we believe that it is still useful for the following two reasons.

(i) The computation time of SVM can be intolerably deteriorated as in Figures 6 and 8, for the Industry Sector and the TechTC-100 datasets, respectively, while the gamma-Poisson classifier offers the second best classification performance with moderate computation times even for these two cases.

(ii) The gamma-Poisson classifier can be conveniently used for a wide range of practical systems in which continuous incremental learning tasks are required.

In the following, we first try to explain the origin of the slow computation times of SVM for the Industry Sector and the TechTC-100 datasets and then describe the effective incremental learning of the gamma-Poisson classifier which is suitable in practical systems.

6.2.1. Slow Computation Time of SVM for Noisy Dataset. In general, the problem is linearly separable for SVM if each term in the vocabulary is almost peculiar to one of the all possible categories and, in this case, the SVM can be easily trained within a short time. Contrary, if many terms (a large portion of the vocabulary) tend to disperse over at least several categories with some finite probabilities, the linear separability decreases and the SVM suffers a long computation time to find an optimum hyperplane. For SVM, a dataset is regarded as noisy if the latter case occurs while the computation times of all the other probabilistic classifiers are not affected by the noisy nature because optimization procedures are not included in these classifiers. Concerning the origin of the noisy nature for the Industry Sector and the TechTC-100 datasets, we can consider following reasons.

(i) These datasets are generated from web pages. As mentioned earlier, the textual advertisements included in the web pages can make term distributions noisy because the same kind of advertisements tend to appear across multiple categories.

(ii) The numbers of classes, 104 for the Industry Sector and 40 for the TechTC-100, are much larger than those of the other two datasets. This causes the noisy nature because the similar concepts (almost similar topic) are nearly equally distributed among several adjacent categories when the categorization has been made in a fine-grained manner for a dataset with a large number of classes.

Figure 10 depicts the noisy nature of the Industry Sector and the TechTC-100 datasets. To obtain the figure, the following procedures were applied.

(1) All the terms in the vocabulary were sorted by the collection term frequency, CF, which is the total frequency of each word throughout the entire dataset.

(2) 2,000 top terms were chosen based on the CF values.

(3) A probability $P(C_{\text{top}})$ defined by

$$P\left(C_{\text{top}}\right) = \frac{\text{frequency of the term in the most relevant category}}{\text{collection term frequency, CF}} \tag{27}$$

was calculated for each of the top 2,000 terms. $P(C_{\text{top}})$ is regarded as the simple probability estimation for the occurrence of a considered term in the most relevant category. Note that, for example, if the considered term has $P(C_{\text{top}})$ less than 20%, then the term is distributed over at least 6 categories.

(4) The top 2,000 terms were sorted by the value of $P(C_{\text{top}})$ and plotted as in Figure 10 where the x-axis shows the respective rank of $P(C_{\text{top}})$ and the y-axis the value of $P(C_{\text{top}})$.

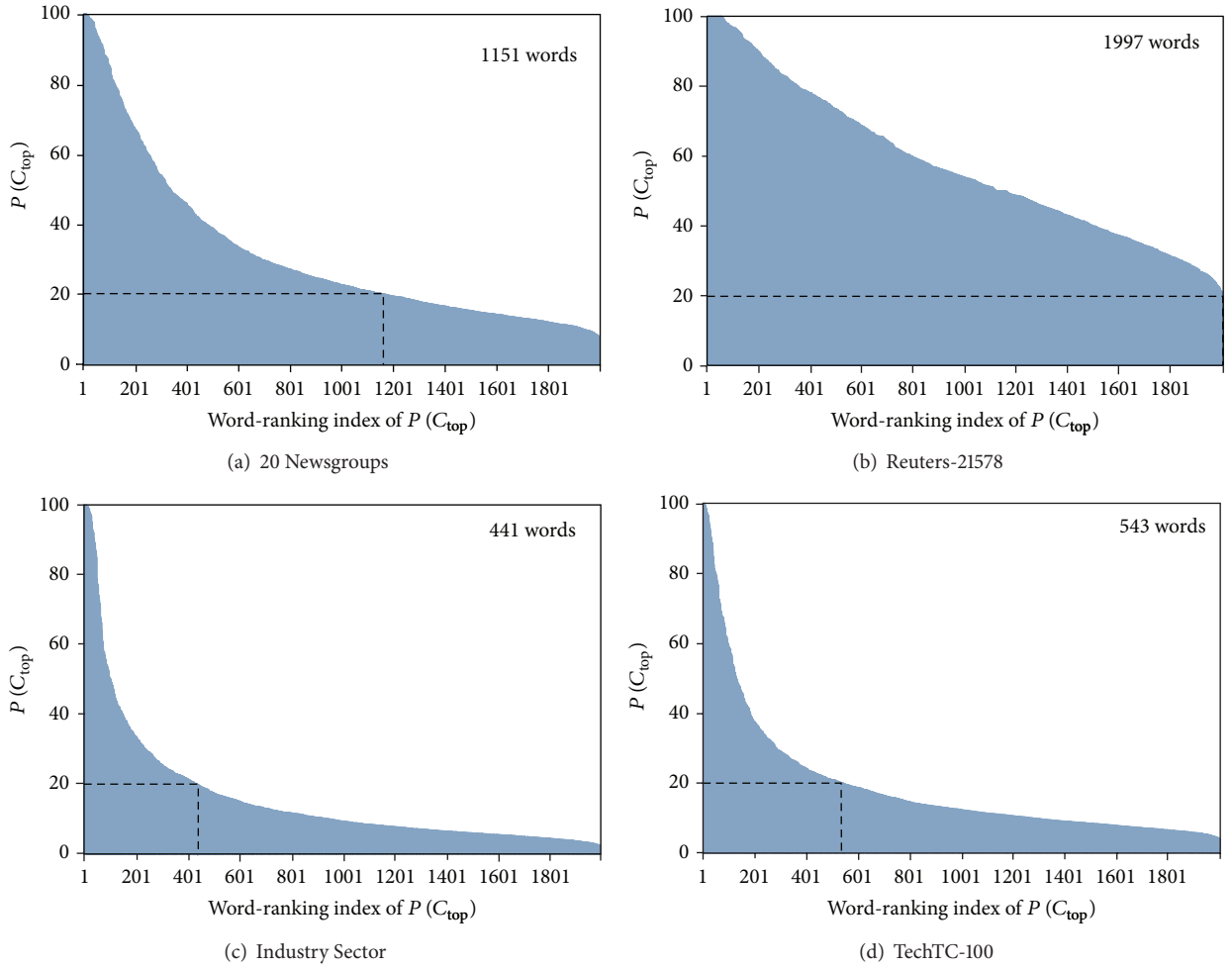
In the figure, we tentatively show the number of terms with $P(C_{\text{top}})$ that is larger than 20%. The result indicates that about three-quarters of the top 2,000 terms are distributed over at least 6 categories for the Industry Sector and the TechTC-100 datasets. Therefore, these two datasets are noisy for SVM in the sense that we have described above compared with the other two datasets.

The performance of SVM for "noisy" data might be improved by using other kernels instead of using linear kernel which is utilized in this study; however, selection of a new kernel and the optimization of kernel parameters are newly introduced as additional tasks in this case. Furthermore, the fast computation time of SVM for a linearly separable dataset as demonstrated in Figures 3 and 4 cannot be expected with other kernels (the $\text{SVM}^{\text{multiclass}}$ is optimized for the linear kernel so that the runtime can be scaled linearly with the number of training examples by use of a cutting-plane algorithm). By contrast, the performance of the gamma-Poisson classifier is stable even in the case of noisy datasets with moderate computation times.

6.2.2. Effective Incremental Learning of the Gamma-Poisson Classifier. As mentioned above, the gamma-Poisson classifier can be used for a wide array of practical systems in which continuous incremental learning tasks are required. Typical examples are spam-filtering and adaptive news-alert systems.

Table 4: Classification results for the 20 Newsgroups dataset. For our results, we show the data obtained with all the words in the initial vocabulary.

Model	Present work	Allison [9]	Madsen et al. [7]	Clinchant and Gaussier [8]	Joachims [21]	Kibriya et al. [22]
Multinomial	0.8934 ± 0.0069	0.8566 ± 0.0050	0.853 ± 0.004	0.875	0.896	0.8836
Dirichlet-multinomial	0.8877 ± 0.0046	0.8503 ± 0.0051	0.890 ± 0.005	0.878		
beta-binomial	0.9143 ± 0.0049	0.9165 ± 0.0040				
gamma-Poisson	0.9148 ± 0.0066					
SVM	0.9144 ± 0.0060	0.8880 ± 0.0045				0.9352



(a) 20 Newsgroups

(b) Reuters-21578

(c) Industry Sector

(d) TechTC-100

Figure 10: Probability of the occurrence of a considered term in the most relevant category, $P(C_{\text{top}})$, for the most frequent 2,000 terms.

In these systems, a small number of new training documents are continuously provided and the retraining of the classifier with the new training documents is routinely needed. In such an environment, the proposed classifier is more appropriate for practical systems rather than other complex learning models including SVM because our classifier ensures effective incremental learning.

To show the effectiveness, we consider a situation in which one new training document belonging to a class c with a document vector $d'_c = (x'_1, x'_2, \ldots, x'_{|V|})$ is supplied to our classifier. In this case, the set of training vectors becomes the union of original training vectors $D_c = \{d_{ci}\}$ $(1 \leq i \leq |D_c|)$ and one newly supplied vector d'_c. The retraining of the classifier corresponds to estimating a new set of parameters

$\{\widehat{\alpha}'_{cj}\}$ and $\{\widehat{\beta}'_{cj}\}$ $(1 \le j \le |V|)$ from the $|D_c|+1$ training vectors. Referring to definitions of $\overline{x}_j$ and $\widetilde{x}_j$ ((15) and (16)), we can easily verify that the new values, $\overline{x}'_j$ and $\widetilde{x}'_j$ for $|D_c|+1$ training vectors, can be expressed in terms of the original $\overline{x}_j$ and $\widetilde{x}_j$ as

$$\begin{aligned}\overline{x}'_j &\equiv \frac{1}{|D_c|+1}\sum_{i=1}^{|D_c|+1} x_{ij} \\ &= \frac{1}{|D_c|+1}\left(|D_c|\,\overline{x}_j + x'_j\right), \\ \ln \widetilde{x}'_j &\equiv \ln\left(\prod_{i=1}^{|D_c|+1} x_{ij}\right)^{1/(|D_c|+1)} \\ &= \frac{1}{|D_c|+1}\left(|D_c|\ln \widetilde{x}_j + \ln x'_j\right).\end{aligned} \tag{28}$$

The last expressions in (28) indicate that we can directly update the values of $\overline{x}_j$ and $\widetilde{x}_j$ from their original values. Because we can use these last expressions for the updates, and because the summation and the product over all training documents seen in the first expressions of (28) are actually unnecessary; the time complexity of retraining for all terms in the vocabulary using the combination of (28), (17), (18) and (19) is estimated to be only $O(|V|)$. This is a clear advantage of gamma-Poisson modeling for incremental learning for the following reasons.

(i) The time complexity of retraining the classifier using beta-binomial modeling in the same situation is given as $O(|V||D_c|)$ because it uses the method of moments for the estimation of parameters. The complexity is much higher than the case of gamma-Poisson modeling.

(ii) The time complexity for retraining the classifier using multinomial modeling is $O(|V|)$, exactly the same as in the case of gamma-Poisson modeling. (If the total counts for each term over original training vectors are stored in the classifier, then the counts are simply updated by adding the corresponding counts in the newly provided training vector and the new set of parameters $\{\widehat{\theta}_{cj}\}$ is immediately obtained by using (4) with these updated counts. This procedure results in complexity of $O(|V|)$.) However, from the comparison of overall performance described above, gamma-Poisson modeling is preferable over multinomial modeling since the former shows better performance.

(iii) In the case of SVM, it has been clarified that the support vectors, which can be regarded as summarized information of original training vectors, are sufficient for incremental learning [24]. Along the line of this framework, the new set of training vectors becomes the union of support vectors obtained from original training vectors and one newly provided training vector; we can thus retrain SVM with this new training set for incremental learning. The time for training an SVM is dominated by the time for solving the underlying quadratic programming, and so the theoretical and empirical complexity varies depending on the method used to solve it. Although the number of support vectors is much smaller than $|D|$, the time for doing quadratic optimization is considered to be much slower than simply counting terms [25] as is done in multinomial and gamma-Poisson modeling.

The last expressions in (28) also imply that storing the set of all training vectors is needless for the incremental learning because only the values of $\{\overline{x}_j\}$ and $\{\widetilde{x}_j\}$ are sufficient for the update. This indicates that the gamma-Poisson classifier is suitable for incremental learning in terms of not only time complexity but also space complexity.

7. Conclusions and Future Work

In this paper, we have proposed a novel classifier in which the gamma-Poisson distribution is utilized as a new tool for text modeling. The gamma-Poisson was introduced in order to improve the insufficient description of word occurrences from the original Poisson distribution. The framework of gamma-Poisson modeling of texts and the construction of a classifier using the framework were demonstrated with practical techniques for parameter estimation and vector normalization. The efficiency of the proposed classifier was examined through experiments on automatic text categorization of the 20 Newsgroups, Reuters-21578, Industry Sector, and TechTC-100 datasets. For comparison, classifiers using three other distributions, namely, multinomial, Dirichlet-multinomial and beta-binomial distributions, were also applied to the same datasets, and in addition we also used SVM as a standard discriminative classifier with state-of-the-art classification accuracy. From the results, it was found that the proposed classifier with the gamma-Poisson model shows classification performance comparable with that of SVM. The origin of the superiority of the proposed gamma-Poisson modeling was discussed in terms of its flexibility in describing various patterns of word distributions and the effectiveness of vector normalization. We also showed that the proposed classifier is most suitable for applications in which continuous incremental learning tasks are required.

At present, the analysis of classification performance for the various classifiers remains unsatisfactory because the results are interpreted only qualitatively. We consider that further quantitative discussion is possible through analysis of the decision functions of the classifiers. An investigation along the line of such quantitative analysis is reserved for future research.

Acknowledgment

The authors would like to acknowledge Dr. Yusuke Higuchi for useful discussion and illuminating suggestions.

References

[1] K. Church and W. A. Gale, "Inverse Document Frequency (IDF): a measure of deviations from poisson," in *Proceedings of the 3rd Workshop on Very Large Corpora*, pp. 121–130, 1995.

[2] H. Ogura, H. Amano, and M. Kondo, "Feature selection with a measure of deviations from Poisson in text categorization," *Expert Systems with Applications*, vol. 36, no. 3, pp. 6826–6832, 2009.

[3] H. Ogura, H. Amano, and M. Kondo, "Distinctive characteristics of a metric using deviations from Poisson for feature selection," *Expert Systems with Applications*, vol. 37, no. 3, pp. 2273–2281, 2010.

[4] H. Ogura, H. Amano, and M. Kondo, "Comparison of metrics for feature selection in imbalanced text classification," *Expert Systems with Applications*, vol. 38, no. 5, pp. 4978–4989, 2011.

[5] K. Church and W. A. Gale, "Poisson mixtures," *Natural Language Engineering*, vol. 1, pp. 163–190, 1995.

[6] A. Gelman, B. Carlin, S. Stern, and B. Rubin, *Bayesian Data Analysis (Texts in Statistical Science)*, Chapman and Hall/CRC, 2nd edition, 2003.

[7] R. E. Madsen, D. Kauchak, and C. Elkan, "Modeling word burstiness using the Dirichlet distribution," in *Proceedings of the 22nd International Conference on Machine Learning (ICML '05)*, pp. 545–552, August 2005.

[8] S. Clinchant and E. Gaussier, "The BNB distribution for text modeling," in *Proceedings of the Advances in Information Retrieval. 30th European Conference on IR Research*, pp. 150–161, 2008.

[9] B. Allison, "An improved hierarchical Bayesian Model of Language for document classification," in *Proceedings of the 22nd International Conference on Computational Linguistics*, pp. 25–32, 2008.

[10] S. Eyheramendy, D. Lewis, and D. Madigam, "On the naive bayes model for text categorization," in *Proceedings of the 9th International Workshop on Artificial Intelligence and Statistics*, pp. 332–339, 2003.

[11] T. Mitchell, *Machine Learning*, McGraw Hill, 1997.

[12] S. Kim, K. Han, H. Rim, and H. Myaeng, "Some effective techniques for naive Bayes text classification," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, pp. 1457–1466, 2006.

[13] T. Minka, "Estimating a Dirichlet distribution," 2003, http://research.microsoft.com/en-us/um/people/minka/papers/dirichlet/.

[14] T. Joachims, *Learning to Classify Text Using Support Vector Machines [Ph.D. thesis]*, Kluwer, 2002.

[15] J. Grim, J. Novovičová, and P. Somol, "Structural Poisson mixtures for classification of documents," in *Proceedings of the 19th International Conference on Pattern Recognition (ICPR '08)*, pp. 1–4, December 2008.

[16] A. Greenwood and D. Durand, "Aids for fitting the Gamma distribution by Maximum Likelihood," *Technometrics*, vol. 2, p. 55, 1960.

[17] T. Minka, "Estimating a Gamma distribution," 2002, http://research.microsoft.com/en-us/um/people/minka/papers/.

[18] J. D. M. Rennie, L. Shih, J. Teevan, and D. Karger, "Tackling the poor assumptions of naive bayes text classifiers," in *Proceedings of the 20th International Conference on Machine Learning*, pp. 616–623, August 2003.

[19] K. Lang, "NewsWeeder: learning to filter netnews," in *Proceedings of the 12th International Machine Learning Conference*, pp. 331–339, Morgan Kaufmann, 1995.

[20] D. Davidov, E. Gabrilovich, and S. Markovitch, "Parameterized generation of labeled datasets for text categorization based on a hierarchical directory," in *Proceedings of Sheffield SIGIR 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 250–257, July 2004.

[21] T. Joachims, "A probabilistic analysis of the Rocchio algorithm with TF-IDF for text categorization," in *Proceedings of the 40th International Conference on Machine Learning*, pp. 143–151, 1997.

[22] A. M. Kibriya, E. Frank, B. Pfahringer, and G. Holmes, "Multinomial naive bayes for text categorization revisited," in *Proceedings of the 17th Australian Joint Conference on Artificial Intelligence*, pp. 488–499, 2005.

[23] N. Slonim, G. Bejerano, S. Fine, and N. Tishby, "Discriminative feature selection via multiclass variable memory Markov model," *Eurasip Journal on Applied Signal Processing*, vol. 2003, no. 2, pp. 93–102, 2003.

[24] N. A. Syed, H. Liu, and K. K. Suang, "Incremental learning with support vector machines," in *Proceedings of the Workshop on Support Vector Machines at the International Joint Conference on Artificial Intelligence*, pp. 317–321, 1999.

[25] C. Manning, P. Raghavan, and H. Schutze, *Introduction to Information Retrieval*, Cambridge University Press, 2008.