A Study of Poisson Query Generation Model for Information Retrieval

Qiaozhu Mei, Hui Fang, Chengxiang Zhai
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana,IL 61801
{qmei2,hfang,czhai}@uiuc.edu

ABSTRACT

Many variants of language models have been proposed for information retrieval. Most existing models are based on multinomial distribution and would score documents based on query likelihood computed based on a query generation probabilistic model. In this paper, we propose and study a new family of query generation models based on Poisson distribution. We show that while in their simplest forms, the new family of models and the existing multinomial models are equivalent, they behave differently for many smoothing methods. We show that the Poisson model has several advantages over the multinomial model, including naturally accommodating per-term smoothing and allowing for more accurate background modeling. We present several variants of the new model corresponding to different smoothing methods, and evaluate them on four representative TREC test collections. The results show that while their basic models perform comparably, the Poisson model can outperform multinomial model with per-term smoothing. The performance can be further improved with two-stage smoothing.

Categories and Subject Descriptors: H.3.3 [Information Search and Retrieval]: Retrieval Models

General Terms: Algorithms

Keywords: Language models, Poisson process, query generation, formal models, term dependent smoothing

1. INTRODUCTION

As a new type of probabilistic retrieval models, language models have been shown to be effective for many retrieval tasks [21, 28, 14, 4]. Among many variants of language models proposed, the most popular and fundamental one is the query-generation language model [21, 13], which leads to the query-likelihood scoring method for ranking documents. In such a model, given a query q and a document d, we compute the likelihood of "generating" query q with a model estimated based on document d, i.e., the conditional probability $p(q|d)$. We can then rank documents based on the likelihood of generating the query.

Virtually all the existing query generation language models are based on either multinomial distribution [19, 6, 28] or multivariate Bernoulli distribution [21, 18]. The multinomial distribution is especially popular and also shown to be quite effective. The heavy use of multinomial distribution is partly due to the fact that it has been successfully used in speech recognition, where multinomial distribution is a natural choice for modeling the occurrence of a particular word in a particular position in text. Compared with multivariate Bernoulli, multinomial distribution has the advantage of being able to model the frequency of terms in the query; in contrast, multivariate Bernoulli only models the presence and absence of query terms, thus cannot capture different frequencies of query terms. However, multivariate Bernoulli also has one potential advantage over multinomial from the viewpoint of retrieval: in a multinomial distribution, the probabilities of all the terms must sum to 1, making it hard to accommodate per-term smoothing, while in a multivariate Bernoulli, the presence probabilities of different terms are completely independent of each other, easily accommodating per-term smoothing and weighting. Note that term absence is also indirectly captured in a multinomial model through the constraint that all the term probabilities must sum to 1.

In this paper, we propose and study a new family of query generation models based on the Poisson distribution. In this new family of models, we model the frequency of each term independently with a Poisson distribution. To score a document, we would first estimate a multivariate Poisson model based on the document, and then score it based on the likelihood of the query given by the estimated Poisson model. In some sense, the Poisson model combines the advantage of multinomial in modeling term frequency and the advantage of the multivariate Bernoulli in accommodating per-term smoothing. Indeed, similar to the multinomial distribution, the Poisson distribution models term frequencies, but without the constraint that all the term probabilities must sum to 1, and similar to multivariate Bernoulli, it models each term independently, thus can easily accommodate per-term smoothing.

As in the existing work on multinomial language models, smoothing is critical for this new family of models. We derive several smoothing methods for Poisson model in parallel to those used for multinomial distributions, and compare the corresponding retrieval models with those based on multi-

SIGIR'07, July 23–27, 2007, Amsterdam, The Netherlands.

nomial distributions. We find that while with some smoothing methods, the new model and the multinomial model lead to exactly the same formula, with some other smoothing methods they diverge, and the Poisson model brings in more flexibility for smoothing. In particular, a key difference is that the Poisson model can naturally accommodate *per-term smoothing*, which is hard to achieve with a multinomial model without heuristic twist of the semantics of a generative model. We exploit this potential advantage to develop a new term-dependent smoothing algorithm for Poisson model and show that this new smoothing algorithm can improve performance over term-independent smoothing algorithms using either Poisson or multinomial model. This advantage is seen for both one-stage and two-stage smoothing. Another potential advantage of the Poisson model is that its corresponding background model for smoothing can be improved through using a mixture model that has a closed form formula. This new background model is shown to outperform the standard background model and reduce the sensitivity of retrieval performance to the smoothing parameter.

The rest of the paper is organized as follows. In Section 2, we introduce the new family of query generation models with Poisson distribution, and present various smoothing methods which lead to different retrieval functions. In Section 3, we analytically compare the Poisson language model with the multinomial language model, from the perspective of retrieval. We then design empirical experiments to compare the two families of language models in Section 4. We discuss the related work in 5 and conclude in 6.

2. QUERY GENERATION WITH POISSON PROCESS

In the query generation framework, a basic assumption is that a query is generated with a model estimated based on a document. In most existing work [12, 6, 28, 29], people assume that each query word is sampled independently from a multinomial distribution. Alternatively, we assume that a query is generated by sampling the frequency of words from a series of independent Poisson processes [20].

2.1 The Generation Process

Let $V = \{w_1, ..., w_n\}$ be a vocabulary set. Let $\mathbf{w}$ be a piece of text composed by an author and $\langle c(w_1), ..., c(w_n)\rangle$ be a frequency vector representing $\mathbf{w}$, where $c(w_i, \mathbf{w})$ is the frequency count of term w_i in text $\mathbf{w}$. In retrieval, $\mathbf{w}$ could be either a query or a document. We consider the frequency counts of the n unique terms in $\mathbf{w}$ as n different types of events, sampled from n independent homogeneous Poisson processes, respectively.

Suppose t is the time period during which the author composed the text. With a homogeneous Poisson process, the frequency count of each event, i.e., the number of occurrences of w_i, follows a Poisson distribution with associated parameter $\lambda_i t$, where λ_i is a rate parameter characterizing the expected number of w_i in a unit time. The probability density function of such a Poisson Distribution is given by

$$P(c(w_i, \mathbf{w}) = k|\lambda_i t) = \frac{e^{-\lambda_i t}(\lambda_i t)^k}{k!}$$

Without losing generality, we set t to the length of the text $\mathbf{w}$ (people write one word in a unit time), i.e., $t = |\mathbf{w}|$.

With n such independent Poisson processes, each explaining the generation of one term in the vocabulary, the likelihood of $\mathbf{w}$ to be generated from such Poisson processes can be written as

$$p(\mathbf{w}|\Lambda) = \prod_{i=1}^{n} p(c(w_i, \mathbf{w})|\Lambda) = \prod_{i=1}^{n} \frac{e^{-\lambda_i \cdot |\mathbf{w}|}(\lambda_i \cdot |\mathbf{w}|)^{c(w_i, \mathbf{w})}}{c(w_i, \mathbf{w})!}$$

where $\Lambda = \{\lambda_1, ..., \lambda_n\}$ and $|\mathbf{w}| = \sum_{i=1}^{n} c(w_i, \mathbf{w})$. We refer to these n independent Poisson processes with parameter Λ as a **Poisson Language Model**.

Let $D = \{d_1, ..., d_m\}$ be an observed set of document samples generated from the Poisson process above. The maximum likelihood estimate (MLE) of λ_i is

$$\hat{\lambda_i} = \frac{\sum_{d \in D} c(w_i, d)}{\sum_{d \in D} \sum_{w' \in V} c(w', d)}$$

Note that this MLE is different from the MLE for the Poisson distribution without considering the document lengths, which appears in [22, 24].

Given a document d, we may estimate a Poisson language model Λ_d using d as a sample. The likelihood that a query q is generated from the document language model Λ_d can be written as

$$p(q|d) = \prod_{w \in V} p(c(w, q)|\Lambda_d) \quad (1)$$

This representation is clearly different from the multinomial query generation model as (1) the likelihood includes all the terms in the vocabulary V, instead of only those appearing in q, and (2) instead of the appearance of terms, the event space of this model is the frequencies of each term.

In practice, we have the flexibility to choose the vocabulary V. In one extreme, we can use the vocabulary of the whole collection. However, this may bring in noise and considerable computational cost. In the other extreme, we may focus on the terms in the query and ignore other terms, but some useful information may be lost by ignoring the non-query terms. As a compromise, we may conflate all the non-query terms as one single pseudo term. In other words, we may assume that there is exactly one "non-query term" in the vocabulary for each query. In our experiments, we adopt this "pseudo non-query term" strategy.

A document can be scored with the likelihood in Equation 1. However, if a query term is unseen in the document, the MLE of the Poisson distribution would assign zero probability to the term, causing the probability of the query to be zero. As in existing language modeling approaches, the main challenge of constructing a reasonable retrieval model is to find a smoothed language model for $p(\cdot|d)$.

2.2 Smoothing in Poisson Retrieval Model

In general, we want to assign non-zero rates for the query terms that are not seen in document d. Many smoothing methods have been proposed for multinomial language models[2, 28, 29]. In general, we have to discount the probabilities of some words seen in the text to leave some extra probability mass to assign to the unseen words. In Poisson language models, however, we do not have the same constraint as in a multinomial model (i.e., $\sum_{w \in V} p(w|d) = 1$). Thus we do not have to discount the probability of seen words in order to give a non-zero rate to an unseen word. Instead, we only need to guarantee that $\sum_{k=0,1,2,...} p(c(w, d) = k|d) = 1$. In this section, we introduce three different strategies to smooth a Poisson language model, and show how they lead to different retrieval functions.

2.2.1 Bayesian Smoothing using Gamma Prior

Following the risk minimization framework in [11], we assume that a document is generated by the arrival of terms in a time period of $|d|$ according to the document language model, which essentially consists of a vector of Poisson rates for each term, i.e., $\Lambda_d = \langle \lambda_{d,1}, ..., \lambda_{d,|V|} \rangle$.

A document is assumed to be generated from a potentially different model. Given a particular document d, we want to estimate Λ_d. The rate of a term is estimated independently of other terms. We use Bayesian estimation with the following Gamma prior, which has two parameters, α and β:

$$Gamma(\lambda|\alpha,\beta) = \frac{\beta^\alpha}{\Gamma(\alpha)}\lambda^{\alpha-1}e^{-\beta\lambda}$$

For each term w, the parameters α_w and β_w are chosen to be $\alpha_w = \mu * \lambda_{C,w}$ and $\beta_w = \mu$, where μ is a parameter and $\lambda_{C,w}$ is the rate of w estimated from some background language model, usually the "collection language model". The posterior distribution of Λ_d is given by

$$p(\Lambda_d|d,C) \propto \prod_{w\in V} e^{-\lambda_w(|d|+\mu)}\lambda_w^{c(w,d)+\mu\lambda_{C,w}-1}$$

which is a product of $|V|$ Gamma distributions with parameters $c(w,d) + \mu\lambda_{C,w}$ and $|d| + \mu$ for each word w. Given that the Gamma mean is $\frac{\alpha}{\beta}$, we have

$$\hat{\lambda}_{d,w} = \int_{\lambda_{\mathbf{d},\mathbf{w}}} \lambda_{d,w}p(\lambda_{d,w}|d,C)\mathbf{d}\lambda_{\mathbf{d},\mathbf{w}} = \frac{c(w,d)+\mu\lambda_{C,w}}{|d|+\mu}$$

This is precisely the smoothed estimate of multinomial language model with Dirichlet prior [28].

2.2.2 Interpolation (Jelinek-Mercer) Smoothing

Another straightforward method is to decompose the query generation model as a mixture of two component models. One is the document language model estimated with maximum likelihood estimator, and the other is a model estimated from the collection background, $p(\cdot|C)$, which assigns non-zero rate to w.

For example, we may use an interpolation coefficient between 0 and 1 (i.e., $\delta \in [0,1]$). With this simple interpolation, we can score a document with

$$Score(d,q) = \sum_{w\in V} \log((1-\delta)p(c(w,q)|d) + \delta p(c(w,q)|C)) \quad (2)$$

Using the maximum likelihood estimator for $p(\cdot|d)$, we have $\lambda_{d,w} = \frac{c(w,d)}{|d|}$, thus Equation 2 becomes

$$\begin{aligned} Score(d,q) \propto & \sum_{w\in d\cap q} [\log(1+\frac{1-\delta}{\delta}\frac{e^{-\lambda_{d,w}|q|}(\lambda_{d,w}|q|)^{c(w,q)}}{c(w,q)!\cdot p(c(w,q)|C)}) \\ & - \log\frac{(1-\delta)e^{-\lambda_{d,w}|q|}+\delta p(c(w,q)=0|C)}{1-\delta+\delta p(c(w,q)=0|C)}] \\ & + \sum_{w\in d}\log\frac{(1-\delta)e^{-\lambda_{d,w}|q|}+\delta p(c(w,q)=0|C)}{1-\delta+\delta p(c(w,q)=0|C)} \end{aligned}$$

We can also use a Poisson language model for $p(\cdot|C)$, or use some other frequency-based models. In the retrieval formula above, the first summation can be computed efficiently. The second summation can be actually treated as a document prior, which penalizes long documents.

As the second summation is difficult to compute efficiently, we conflate all non-query terms as one pseudo "non-query-term", denoted as "N". Using the pseudo-term formulation and a Poisson collection model, we can rewrite the retrieval formula as

$$\begin{aligned} Score(d,q) \propto & \sum_{w\in d\cap q}\log(1+\frac{1-\delta}{\delta}\frac{e^{-\lambda_{d,w}}(\lambda_{d,w}|q|)^{c(w,q)}}{e^{-\lambda_{d,C}|q|}(\lambda_{d,C})^{c(w,q)}}) \\ & + \log\frac{(1-\delta)e^{-\lambda_{d,N}|q|}+\delta e^{-\lambda_{C,N}|q|}}{1-\delta+\delta e^{-\lambda_{C,N}|q|}} \end{aligned} \quad (3)$$

where $\lambda_{d,N} = \frac{|d|-\sum_{w\in q}c(w,d)}{|d|}$ and $\lambda_{C,N} = \frac{|C|-\sum_{w\in q}c(w,C)}{|C|}$.

2.2.3 Two-Stage Smoothing

As discussed in [29], smoothing plays two roles in retrieval: (1) to improve the estimation of the document language model, and (2) to explain the common terms in the query. In order to distinguish the content and non-discriminative words in a query, we follow [29] and assume that a query is generated by sampling from a two-component mixture of Poisson language models, with one component being the document model Λ_d and the other being a query background language model $p(\cdot|U)$. $p(\cdot|U)$ models the "typical" term frequencies in the user's queries. We may then score each document with the query likelihood computed using the following two-stage smoothing model:

$$p(c(w,q)|\Lambda_d,U) = (1-\delta)p(c(w,q)|\Lambda_d) + \delta p(c(w,q)|U) \quad (4)$$

where δ is a parameter, roughly indicating the amount of "noise" in q. This looks similar to the interpolation smoothing, except that $p(\cdot|\Lambda_d)$ now should be a smoothed language model, instead of the one estimated with MLE.

With no prior knowledge on $p(\cdot|U)$, we could set it to $p(\cdot|C)$. Any smoothing methods for the document language model can be used to estimate $p(\cdot|d)$ such as the Gamma smoothing as discussed in Section 2.2.1.

The empirical study of the smoothing methods is presented in Section 4.

3. ANALYSIS OF POISSON LANGUAGE MODEL

From the previous section, we notice that the Poisson language model has a strong connection to the multinomial language model. This is expected since they both belong to the exponential family [26]. However, there are many differences when these two families of models are applied with different smoothing methods. From the perspective of retrieval, will these two language models perform equivalently? If not, which model provides more benefits to retrieval, or provides flexibility which could lead to potential benefits? In this section, we analytically discuss the retrieval features of the Poisson language models, by comparing their behavior with that of the multinomial language models.

3.1 The Equivalence of Basic Models

Let us begin with the assumption that all the query terms appear in every document. Under this assumption, no smoothing is needed. A document can be scored by the log likelihood of the query with the maximum likelihood estimate:

$$Score(d,q) = \sum_{w\in V}\log\frac{e^{-\lambda_{d,w}|q|}(\lambda_{d,w}|q|)^{c(w,q)}}{c(w,q)!} \quad (5)$$

Using the MLE, we have $\lambda_{d,w} = \frac{c(w,d)}{\sum_{w \in V} c(w,d)}$. Thus

$$Score(d,q) \propto \sum_{c(w,q)>0} c(w,q) \log \frac{c(w,d)}{\sum_{w \in V} c(w,d)}$$

This is exactly the log likelihood of the query if the document language model is a multinomial with maximum likelihood estimate. Indeed, even with Gamma smoothing, when plugging $\lambda_{d,w} = \frac{c(w,d)+\mu\lambda_{C,w}}{|d|+\mu}$ and $\lambda_{C,w} = \frac{c(w,C)}{|C|}$ into Equation 5, it is easy to show that

$$Score(d,q) \propto \sum_{w \in q \cap d} c(w,q) \log(1 + \frac{c(w,d)}{\mu \cdot \frac{c(w,C)}{|C|}}) + |q| \log \frac{\mu}{|d|+\mu} \quad (6)$$

which is exactly the Dirichlet retrieval formula in [28]. Note that this equivalence holds only when the document length variation is modeled with Poisson process.

This derivation indicates the equivalence of the basic Poisson and multinomial language models for retrieval. With other smoothing strategies, however, the two models would be different. Nevertheless, with this equivalence in basic models, we could expect that the Poisson language model performs comparably to the multinomial language model in retrieval, if only simple smoothing is explored. Based on this equivalence analysis, one may ask, why we should pursue the Poisson language model. In the following sections, we show that despite the equivalence in their basic models, the Poisson language model brings in extra flexibility for exploring advanced techniques on various retrieval features, which could not be achieved with multinomial language models.

3.2 Term Dependent Smoothing

One flexibility of the Poisson language model is that it provides a natural framework to accommodate term dependent (per-term) smoothing. Existing work on language model smoothing has already shown that different types of queries should be smoothed differently according to how discriminative the query terms are. [7] also predicted that different terms should have a different smoothing weights. With multinomial query generation models, people usually use a single smoothing coefficient to control the combination of the document model and the background model [28, 29]. This parameter can be made specific for different queries, but always has to be a constant for all the terms. This is mandatory since a multinomial language model has the constraint that $\sum_{w \in V} p(w|d) = 1$. However, from retrieval perspective, different terms may need to be smoothed differently even if they are in the same query. For example, a non-discriminative term (e.g., "the", "is") is expected to be explained more with the background model, while a content term (e.g., "retrieval", "bush") in the query should be explained with the document model. Therefore, a better way of smoothing would be to set the interpolation coefficient (i.e., δ in Formula 2 and Formula 3) specifically for each term. Since the Poisson language model does not have the "sum-to-one" constraint across terms, it can easily accommodate per-term smoothing without needing to heuristically twist the semantics of a generative model as in the case of multinomial language models. Below we present a possible way to explore term dependent smoothing with Poisson language models.

Essentially, we want to use a term-specific smoothing coefficient δ in the linear combination, denoted as δ_w. This coefficient should intuitively be larger if w is a common word and smaller if it is a content word. The key problem is to find a method to assign reasonable values to δ_w. Empirical tuning is infeasible for so many parameters. We may instead estimate the parameters "$\Delta = \{\delta_1, ..., \delta_{|V|}\}$" by maximizing the likelihood of the query given the mixture model of $p(q|\Lambda_Q)$ and $p(q|U)$, where Λ_Q is the "true" query model to generate the query and $p(q|U)$ is a query background model as discussed in Section 2.2.3.

With the model $p(q|\Lambda_Q)$ hidden, the query likelihood is

$$p(q|\Delta, U) = \int_{\Lambda_Q} \prod_{w \in V} ((1-\delta_w)p(c(w,q)|\Lambda_Q) + \delta_w p(c(w,q)|U)) P(\Lambda_Q|U) d\Lambda_Q$$

If we have relevant documents for each query, we can approximate the query model space with the language models of all the relevant documents. Without relevant documents, we opt to approximate the query model space with the models of all the documents in the collection. Setting $p(\cdot|U)$ as $p(\cdot|C)$, the query likelihood becomes

$$p(q|\Delta, U) = \sum_{d \in C} \pi_d \prod_{w \in V} ((1-\delta_w)p(c(w,q)|\hat{\Lambda}_d) + \delta_w p(c(w,q)|C))$$

where $\pi_d = p(\hat{\Lambda}_d|U)$. $p(\cdot|\hat{\Lambda}_d)$ is an estimated Poisson language model for document d.

If we have prior knowledge on $p(\hat{\Lambda}_d|U)$, such as which documents are relevant to the query, we can set π_d accordingly, because what we want is to find Δ that can maximize the likelihood of the query given relevant documents. Without this prior knowledge, we can leave π_d as free parameters, and use the EM algorithm to estimate π_d and Δ. The updating functions are given as

$$\pi_d^{(k+1)} = \frac{\pi_d \prod_{w \in V} ((1-\delta_w)p(c(w,q)|\hat{\Lambda}_d) + \delta_w p(c(w,q)|C))}{\sum_{d \in C} \pi_d \prod_{w \in V} ((1-\delta_w)p(c(w,q)|\hat{\Lambda}_d) + \delta_w p(c(w,q)|C))}$$

and

$$\delta_w^{(k+1)} = \sum_{d \in C} \pi_d \frac{\delta_w p(c(w,q)|C))}{(1-\delta_w)p(c(w,q)|\hat{\Lambda}_d) + \delta_w p(c(w,q)|C))}$$

As discussed in [29], we only need to run the EM algorithm for several iterations, thus the computational cost is relatively low. We again assume our vocabulary containing all query terms plus a pseudo non-query term. Note that the function does not give an explicit way of estimating the coefficient for the unseen non-query term. In our experiments, we set it to the average over δ_w of all query terms.

With this flexibility, we expect Poisson language models could improve the retrieval performance, especially for verbose queries, where the query terms have various discriminative values. In Section 4, we use empirical experiments to prove this hypothesis.

3.3 Mixture Background Models

Another flexibility is to explore different background (collection) models (i.e., $p(\cdot|U)$, or $p(\cdot|C)$). One common assumption made in language modeling information retrieval is that the background model is a homogeneous model of the document models [28, 29]. Similarly, we can also make the assumption that the collection model is a Poisson language model, with the rates $\lambda_{C,w} = \frac{\sum_{d \in C} c(w,d)}{|C|}$. However, this assumption usually does not hold, since the collection is far more complex than a single document. Indeed, the

collection usually consists of a mixture of documents with various genres, authors, and topics, etc. Treating the collection model as a mixture of document models, instead of a single "pseudo-document model" is more reasonable. Existing work of multinomial language modeling has already shown that a better modeling of background improves the retrieval performance, such as clusters [15, 10], neighbor documents [25], and aspects [8, 27]. All the approaches can be easily adopted using Poisson language models. However, a common problem of these approaches is that they all require heavy computation to construct the background model. With Poisson language modeling, we show that it is possible to model the mixture background without paying for the heavy computational cost.

Poisson Mixture [3] has been proposed to model a collection of documents, which can fit the data much better than a single Poisson. The basic idea is to assume that the collection is generated from a mixture of Poisson models, which has the general form of

$$p(x = k|PM) = \int_{\lambda} p(\lambda)p(x = k|\lambda)d\lambda$$

$p(\cdot|\lambda)$ is a single Poisson model and $p(\lambda)$ is an arbitrary probability density function. There are three well known Poisson mixtures [3]: 2-Poisson, Negative Binomial, and the Katz's K-Mixture [9]. Note that the 2-Poisson model has actually been explored in probabilistic retrieval models, which led to the well-known BM25 formula [22].

All these mixtures have closed forms, and can be estimated from the collection of documents efficiently. This is an advantage over the multinomial mixture models, such as PLSI [8] and LDA [1], for retrieval. For example, the probability density function of Katz's K-Mixture is given as

$$p(c(w) = k|\alpha_w, \beta_w) = (1 - \alpha_w)\eta_{k,0} + \frac{\alpha_w}{\beta_w + 1}(\frac{\beta_w}{\beta_w + 1})^k$$

where $\eta_{k,0} = 1$ when $k = 0$, and 0 otherwise.

With the observation of a collection of documents, α_w and β_w can be estimated as

$$\beta_w = \frac{cf(w) - df(w)}{df(w)} \quad and \quad \alpha_w = \frac{cf(w)}{N\beta_w}$$

where $cf(w)$ and $df(w)$ are the collection frequency and document frequency of w, and N is the number of documents in the collection. To account for the different document lengths, we assume that β_w is a reasonable estimation for generating a document of the average length, and use $\beta' = \frac{\beta_w}{avdl}|q|$ to generate the query. This Poisson mixture model can be easily used to replace $P(\cdot|C)$ in the retrieval functions 3 and 4.

3.4 Other Possible Flexibilities

In addition to term dependent smoothing and efficient mixture background, a Poisson language model has also some other potential advantages. For example, in Section 2, we see that Formula 2 introduces a component which does document length penalization. Intuitively, when the document has more unique words, it will be penalized more. On the other hand, if a document is exactly n copies of another document, it would not get over penalized. This feature is desirable and not achieved with the Dirichlet model [5]. Potentially, this component could penalize a document according to what types of terms it contains. With term specific settings of δ, we could get even more flexibility for document length normalization.

Pseudo-feedback is yet another interesting direction where the Poission model might be able to show its advantage. With model-based feedback, we could again relax the combination coefficients of the feedback model and the background model, and allow different terms to contribute differently to the feedback model. We could also utilize the "relevant" documents to learn better per-term smoothing coefficients.

4. EVALUATION

In Section 3, we analytically compared the Poisson language models and multinomial language models from the perspective of query generation and retrieval. In this section, we compare these two families of models empirically. Experiment results show that the Poisson model with per-term smoothing outperforms multinomial model, and the performance can be further improved with two-stage smoothing. Using Poisson mixture as background model also improves the retrieval performance.

4.1 Datasets

Since retrieval performance could significantly vary from one test collection to another, and from one query to another, we select four representative TREC test collections: AP, Trec7, Trec8, and Wt2g(Web). To cover different types of queries, we follow [28, 5], and construct short-keyword (SK, keyword title), short-verbose (SV, one sentence description), and long-verbose (LV, multiple sentences) queries. The documents are stemmed with the Porter's stemmer, and we do not remove any stop word. For each parameter, we vary its value to cover a reasonably wide range.

4.2 Comparison to Multinomial

We compare the performance of the Poisson retrieval models and multinomial retrieval models using interpolation (Jelinek-Mercer, JM) smoothing and Bayesian smoothing with conjugate priors. Table 1 shows that the two JM-smoothed models perform similarly on all data sets. Since the Dirichlet Smoothing for multinomial language model and the Gamma Smoothing for Poisson language model lead to the same retrieval formula, the performance of these two models are jointly presented. We see that Dirichlet/Gamma smoothing methods outperform both Jelinek-Mercer smoothing methods. The parameter sensitivity curves for two Jelinek-Mercer

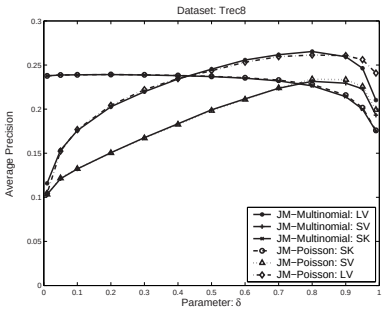

Figure 1: Poisson and multinomial performs similarly with Jelinek-Mercer smoothing

smoothing methods are shown in Figure 1. Clearly, these two methods perform similarly either in terms of optimality

Data	Query	JM-Multinomial			JM-Poisson			Dirichlet/Gamma			Per-term 2-Stage Poisson		
		MAP	InitPr	Pr@5d	MAP	InitPr	Pr@5d	MAP	InitPr	Pr@5d	MAP	InitPr	Pr@5d
AP88-89	SK	0.203	0.585	0.356	0.203	0.585	0.358	0.224	0.629	0.393	**0.226**	0.630	0.396
	SV	0.187	0.580	0.361	0.183	0.571	0.345	0.204	0.613	0.387	**0.217***	0.603	0.390
	LV	0.283	0.716	0.480	0.271	0.692	0.470	0.291	0.710	0.496	**0.304***	0.695	0.510
Trec7	SK	0.167	0.635	0.400	0.168	0.635	0.404	0.186	0.687	0.428	0.185	0.646	0.436
	SV	0.174	0.655	0.432	0.176	0.653	0.432	0.182	0.666	0.432	**0.196***	0.660	0.440
	LV	0.223	0.730	0.496	0.215	0.766	0.488	0.224	0.748	0.52	**0.236***	0.738	0.512
Trec8	SK	0.239	0.621	0.440	0.239	0.621	0.436	0.257	0.718	0.496	0.256	0.704	0.468
	SV	0.231	0.686	0.448	0.234	0.702	0.456	0.228	0.691	0.456	**0.246***	0.692	0.476
	LV	0.265	0.796	0.548	0.261	0.757	0.520	0.260	0.741	0.492	**0.274***	0.766	0.508
Web	SK	0.250	0.616	0.380	0.250	0.616	0.380	0.302	0.767	0.468	**0.307**	0.739	0.468
	SV	0.214	0.611	0.392	0.217	0.609	0.384	0.273	0.693	0.508	**0.292***	0.703	0.480
	LV	0.266	0.790	0.464	0.259	0.776	0.452	0.283	0.756	0.496	**0.311***	0.759	0.488

Table 1: Performance comparison between Poisson and Multinomial retrieval models: basic models perform comparably; term dependent two-stage smoothing significantly improves Poisson
An asterisk (*) indicates that the difference between the performance of the term dependent two-stage smoothing and that of the Dirichlet/Gamma single smoothing is statistically significant according to the Wilcoxon signed rank test at the level of 0.05.

or sensitivity. This similarity of performance is expected as we discussed in Section 3.1.

Although the Poisson model and multinomial model are similar in terms of the basic model and/or with simple smoothing methods, the Poisson model has great potential and flexibility to be further improved. As shown in the rightmost column of Table 1, term dependent two-stage Poisson model consistently outperforms the basic smoothing models, especially for verbose queries. This model is given in Formula 4, with a Gamma smoothing for the document model $p(\cdot|d)$, and δ_w, which is term dependent. The parameter μ of the first stage Gamma smoothing is empirically tuned. The combination coefficients (i.e., Δ), are estimated with the EM algorithm in Section 3.2. The parameter sensitivity curves for Dirichlet/Gamma and the per-term two-stage smoothing model are plotted in Figure 2. The per-term two-stage smoothing method is less sensitive to the parameter μ than Dirichlet/Gamma, and yields better optimal performance.

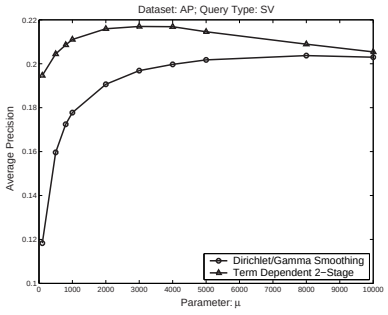

Figure 2: Term dependent two-stage smoothing of Poisson outperforms Dirichlet/Gamma

In the following subsections, we conduct experiments to demonstrate how the flexibility of the Poisson model could be utilized to achieve better performance, which we cannot achieve with multinomial language models.

4.3 Term Dependent Smoothing

To test the effectiveness of the term dependent smoothing, we conduct the following two experiments. In the first experiment, we relax the constant coefficient in the simple Jelinek-Mercer smoothing formula (i.e., Formula 3), and use the EM algorithm proposed in Section 3.2 to find a δ_w for each unique term. Since we are using the EM algorithm to iteratively estimate the parameters, we usually do not want the probability of $p(\cdot|d)$ to be zero. We then use a simple Laplace method to slightly smooth the document model before it goes into the EM iterations. The documents are then still scored with Formula 3, but using learnt δ_w. The results are labeled with "JM+L." in Table 2.

Data	Q	JM	JM	JM+L.	2-Stage	2-Stage
(MAP)	PT:	No	Yes	Yes	No	Yes
AP	SK	0.203	0.204	**0.206**	0.223	**0.226***
	SV	0.183	0.189	**0.214***	0.204	**0.217***
Trec7	SK	0.168	0.171	**0.174**	**0.186**	0.185
	SV	0.176	0.147	**0.198***	0.194	**0.196**
Trec8	SK	0.239	**0.240**	0.227*	**0.257**	0.256
	SV	0.234	0.223	**0.249***	0.242	**0.246***
Web	SK	**0.250**	0.236	0.220*	0.291	**0.307***
	SV	0.217	0.232	**0.261***	0.273	**0.292***

Table 2: Term dependent smoothing improves retrieval performance
An asterisk (*) in Column 3 indicates that the difference between the "JM+L." method and JM method is statistically significant; an asterisk (*) in Column 5 means that the difference between term dependent two-stage method and query dependent two-stage method is statistically significant; PT stands for "per-term".

With term dependent coefficients, the performance of the Jelinek-Mercer Poisson model is improved in most cases. However, in some cases (e.g., Trec7/SV), it performs poorly. This might be caused by the problem of EM estimation with unsmoothed document models. Once non-zero probability is assigned to all the terms before entering the EM iteration, the performance on verbose queries can be improved significantly. This indicates that there is still room to find better methods to estimate δ_w. Please note that neither the per-term JM method nor the "JM+L." method has a parameter to tune.

As shown in Table 1, the term dependent two-stage smoothing can significantly improve retrieval performance. To understand whether the improvement is contributed by the term dependent smoothing or the two-stage smoothing framework, we design another experiment to compare the per-term two-stage smoothing with the two-stage smoothing method proposed in [29]. Their method managed to find coefficients specific to the query, thus a verbose query would use a higher δ. However, since their model is based on multinomial language modeling, they could not get per-term coefficients. We adopt their method to the Poisson two-stage

smoothing, and also estimate a per-query coefficient for all the terms. We compare the performance of such a model with the per-term two-stage smoothing model, and present the results in the right two columns in Table 2. Again, we see that the "per-term" two-stage smoothing outperforms the "per-query" two-stage smoothing, especially for verbose queries. The improvement is not as large as how the per-term smoothing method improves over Dirichlet/Gamma. This is expected, since the per-query smoothing has already addressed the query discrimination problem to some extent. This experiment shows that even if the smoothing is already per-query, making it per-term is still beneficial. In brief, the per-term smoothing improved the retrieval performance of both one-stage and two-stage smoothing method.

4.4 Mixture Background Model

In this section, we conduct experiments to examine the benefits of using a mixture background model without extra computational cost, which can not be achieved for multinomial models. Specifically, in retrieval formula 3, instead of using a single Poisson distribution to model the background $p(\cdot|C)$, we use Katz's K-Mixture model, which is essentially a mixture of Poisson distributions. $p(\cdot|C)$ can be computed efficiently with simple collection statistics, as discussed in Section 3.3.

Data	Query	JM. Poisson	JM. K-Mixture
AP	SK	0.203	**0.204**
	SV	0.183	**0.188***
Trec-7	SK	0.168	**0.169**
	SV	0.176	**0.178***
Trec-8	SK	0.239	0.239
	SV	0.234	**0.238***
Web	SK	0.250	0.250
	SV	0.217	**0.223***

Table 3: K-Mixture background model improves retrieval performance

The performance of the JM retrieval model with single Poisson background and with Katz's K-Mixture background model is compared in Table 3. Clearly, using K-Mixture to model the background model outperforms the single Poisson background model in most cases, especially for verbose queries where the improvement is statistically significant.

Figure 3 shows that the performance changes over different parameters for short verbose queries. The model using K-Mixture background is less sensitive than the one using single Poisson background. Given that this type of mixture

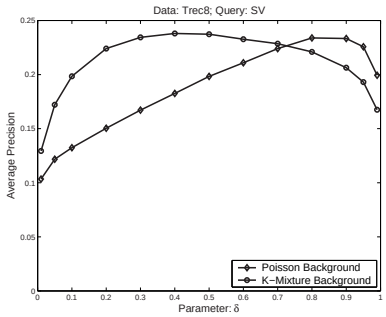

Figure 3: K-Mixture background model deviates the sensitivity of verbose queries

background model does not require any extra computation cost, it would be interesting to study whether using other mixture Poisson models, such as 2-Poisson and negative Binomial, could help the performance.

5. RELATED WORK

To the best of our knowledge, there has been no study of query generation models based on Poisson distribution.

Language models have been shown to be effective for many retrieval tasks [21, 28, 14, 4]. The most popular and fundamental one is the query-generation language model [21, 13]. All existing query generation language models are based on either multinomial distribution [19, 6, 28, 13] or multivariate Bernoulli distribution [21, 17, 18]. We introduce a new family of language models, based on Poisson distribution. Poisson distribution has been previously studied in the document generation models [16, 22, 3, 24], leading to the development of one of the most effective retrieval formula BM25 [23]. [24] studies the parallel derivation of three different retrieval models which is related to our comparison of Poisson and multinomial. However, the Poisson model in their paper is still under the document generation framework, and also does not account for the document length variation. [26] introduces a way to empirically search for an exponential model for the documents. Poisson mixtures [3] such as 2-Poisson [22], Negative multinomial, and Katz's K-Mixture [9] has shown to be effective to model and retrieve documents. Once again, none of this work explores Poisson distribution in the query generation framework.

Language model smoothing [2, 28, 29] and background structures [15, 10, 25, 27] have been studied with multinomial language models. [7] analytically shows that term specific smoothing could be useful. We show that Poisson language model is natural to accommodate the per-term smoothing without heuristic twist of the semantics of a generative model, and is able to efficiently better model the mixture background, both analytically and empirically.

6. CONCLUSIONS

We present a new family of query generation language models for retrieval based on Poisson distribution. We derive several smoothing methods for this family of models, including single-stage smoothing and two-stage smoothing. We compare the new models with the popular multinomial retrieval models both analytically and experimentally. Our analysis shows that while our new models and multinomial models are equivalent under some assumptions, they are generally different with some important differences. In particular, we show that Poisson has an advantage over multinomial in naturally accommodating per-term smoothing. We exploit this property to develop a new per-term smoothing algorithm for Poisson language models, which is shown to outperform term-independent smoothing for both Poisson and multinomial models. Furthermore, we show that a mixture background model for Poisson can be used to improve the performance and robustness over the standard Poisson background model. Our work opens up many interesting directions for further exploration in this new family of models. Further exploring the flexibilities over multinomial language models, such as length normalization and pseudo-feedback could be good future work. It is also appealing to find robust methods to learn the per-term smoothing coefficients without additional computation cost.

7. ACKNOWLEDGMENTS

We thank the anonymous SIGIR 07 reviewers for their useful comments. This material is based in part upon work supported by the National Science Foundation under award numbers IIS-0347933 and 0425852.

8. REFERENCES

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

[2] S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Harvard University, 1998.

[3] K. Church and W. Gale. Poisson mixtures. *Nat. Lang. Eng.*, 1(2):163–190, 1995.

[4] W. B. Croft and J. Lafferty, editors. *Language Modeling and Information Retrieval*. Kluwer Academic Publishers, 2003.

[5] H. Fang, T. Tao, and C. Zhai. A formal study of information retrieval heuristics. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 49–56, 2004.

[6] D. Hiemstra. *Using Language Models for Information Retrieval*. PhD thesis, University of Twente, Enschede, Netherlands, 2001.

[7] D. Hiemstra. Term-specific smoothing for the language modeling approach to information retrieval: the importance of a query term. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 35–41, 2002.

[8] T. Hofmann. Probabilistic latent semantic indexing. In *Proceedings of ACM SIGIR'99*, pages 50–57, 1999.

[9] S. M. Katz. Distribution of content words and phrases in text and language modelling. *Nat. Lang. Eng.*, 2(1):15–59, 1996.

[10] O. Kurland and L. Lee. Corpus structure, language models, and ad hoc information retrieval. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 194–201, 2004.

[11] J. Lafferty and C. Zhai. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of SIGIR'01*, pages 111–119, Sept 2001.

[12] J. Lafferty and C. Zhai. Probabilistic IR models based on query and document generation. In *Proceedings of the Language Modeling and IR workshop*, pages 1–5, May 31 – June 1 2001.

[13] J. Lafferty and C. Zhai. Probabilistic relevance models based on document and query generation. In W. B. Croft and J. Lafferty, editors, *Language Modeling and Information Retrieval*. Kluwer Academic Publishers, 2003.

[14] V. Lavrenko and B. Croft. Relevance-based language models. In *Proceedings of SIGIR'01*, pages 120–127, Sept 2001.

[15] X. Liu and W. B. Croft. Cluster-based retrieval using language models. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 186–193, 2004.

[16] E. L. Margulis. Modelling documents with multiple poisson distributions. *Inf. Process. Manage.*, 29(2):215–227, 1993.

[17] A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification. In *Proceedings of AAAI-98 Workshop on Learning for Text Categorization*, 1998.

[18] D. Metzler, V. Lavrenko, and W. B. Croft. Formal multiple-bernoulli models for language modeling. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 540–541, 2004.

[19] D. H. Miller, T. Leek, and R. Schwartz. A hidden Markov model information retrieval system. In *Proceedings of the 1999 ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 214–221, 1999.

[20] A. Papoulis. *Probability, random variables and stochastic processes*. New York: McGraw-Hill, 1984, 2nd ed., 1984.

[21] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 275–281, 1998.

[22] S. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *Proceedings of SIGIR'94*, pages 232–241, 1994.

[23] S. E. Robertson, S. Walker, S. Jones, M. M.Hancock-Beaulieu, and M. Gatford. Okapi at TREC-3. In D. K. Harman, editor, *The Third Text REtrieval Conference (TREC-3)*, pages 109–126, 1995.

[24] T. Roelleke and J. Wang. A parallel derivation of probabilistic information retrieval models. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 107–114, 2006.

[25] T. Tao, X. Wang, Q. Mei, and C. Zhai. Language model information retrieval with document expansion. In *Proceedings of HLT/NAACL 2006*, pages 407–414, 2006.

[26] J. Teevan and D. R. Karger. Empirical development of an exponential probabilistic model for text retrieval: using textual analysis to build a better model. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 18–25, 2003.

[27] X. Wei and W. B. Croft. Lda-based document models for ad-hoc retrieval. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 178–185, 2006.

[28] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of ACM SIGIR'01*, pages 334–342, Sept 2001.

[29] C. Zhai and J. Lafferty. Two-stage language models for information retrieval. In *Proceedings of ACM SIGIR'02*, pages 49–56, Aug 2002.