Examining document model residuals to provide feedback during Information Retrieval evaluation

Laurence A. F. Park

School of Computing and Mathematics
University of Western Sydney, Australia

lapark@scm.uws.edu.au

Abstract *Evaluation of document models for text based Information retrieval is crucial for developing document models that are appropriate for specific domains. Unfortunately, current document model evaluation methods for text retrieval provide no feedback, except for an evaluation score. To improve a model, we must use trial and error. In this article, we examine how we can provide feedback in the document model evaluation process, by providing a method of computing relevance score residuals and document model residuals for a given document-query set. Document model residuals provide us with an indication of where the document model is accurate and where it is not. We derive a simple method of computing the document model residuals using ridge regression. We also provide an analysis of the residuals of two document models, and show how we can use the correlation of document statistics to the residuals to provide statistically significant improvements to the precision of the model.*

Keywords Information Retrieval, Evaluation, Residuals, Optimisation

1 Introduction

The goal of a text based information retrieval system is to locate text documents, that are relevant to a users query, from a document collection. Research into text based retrieval systems is important since text documents are used in many domains to keep records and store information.

By examining the recent TREC tracks[1], we find uses for text retrieval in the legal and medical domains, general search for the Web and specialised search for homepage finding, blogs, and microblogs. There are many applications for text based information retrieval, and each application is defined by the document collections used and the probability of each query being issued. A specific domain (such as medicine) will require search on a specific document collection, and have a specific distribution over the set of possible queries.

[1] `http://trec.nist.gov/tracks.html`

Proceedings of the 16th Australasian Document Computing Symposium, Canberra, Australia, 2 December 2011.

To evaluate a retrieval system for a specific domain, we should obtain the document collection being used, all possible queries that can be issued, and the probability of each query being issued. The system would be evaluated as the expected evaluation score from a randomly sampled query. Using this form of evaluation, we can obtain the document model weights Δ that best suits the data and queries, by optimising to obtain the best expected evaluation score:

$$\Delta = \arg\max_{D} \sum_{\vec{q}} \text{eval}\left(\vec{r}_q, \vec{s}_q\right) P(q)$$
$$\text{such that } \vec{s}_q = \text{relevance}\left(D, \vec{q}\right)$$

where D is a matrix containing document models weights as its rows, $\vec{q}$ is a query vector containing the query weights for query q, $\vec{r}_q$ is the vector of manual relevance judgements for query q, $\vec{s}_q$ is the vector of document model relevance scores for query q, $P(q)$ is the probability of sampling query q, eval is the evaluation function, and relevance is the document model function used to compute the relevance of the document matrix to the query vector. Unfortunately, this form of evaluation is impossible, since it is unlikely that we would have a database of every possible query, and it would take a huge effort to obtain all the manual relevance judgements for all possible queries on each document in the collection.

The current approach to building an appropriate text retrieval system for a particular domain is to approximate to the expected evaluation score using a small random sample of queries from the domain. Obtaining a small random sample of queries means that we only need a small number of manual relevance judgements, reducing the work required for evaluation. Unfortunately, document models have a large number of parameters (one for each unique term in the document collection), therefore, we are unable to optimise D using a small sample. Doing so would produce a model with zero degrees of freedom and hence overfit the document model to the sample query set.

Therefore, document models are constructed as functions of the document collection and the term frequencies within the documents. By doing this, we are able to build models that generalise well to new query sets. The document models are constructed to approximate the ideal document model weights in Δ. So far, the most popular document models [3, 8, 9, 6] have been constructed based on the distribution

of relevant and irrelevant terms, and by observing statistics such as document model lengths and term counts.

Once these document models are built, we must still ask how can we evaluate them? The current approach to evaluation is to compute the sample mean evaluation score over the sample set of queries. This allows us to compare document models for accuracy, but gives us no insight as to where the document models are performing well and where they are not. We also cannot evaluate how well each model approximates the ideal document model weights Δ for the domain.

We are able to compute the ideal document model by finding the document model that provides the best precision for a given query set, using a set of manual relevance judgements of each query to each document. But, due to the small sample of queries we use, there are many solutions to this problem, and hence it is not obvious which solution we should be comparing our document model to.

In this article, we investigate how we can examine where document models are performing well and where they can be improved, by computing the ideal document model with smallest deviation to the provided document model. Once we have the ideal model, we can analyse the document model residuals. The document model residuals provide us with a means of estimating the accuracy of our document model, and also provide us with feedback as to where the document model deficiencies are for the given document domain. We provide the following contributions:

- a simple method for computing relevance score residuals
- a method of using ridge regression for identifying the ideal document model, and hence the document model residuals
- an analysis of the residuals of the term frequency and BM25 document models using the CRAN document collection.

This article proceeds as follows: Section 2 presents the current form of evaluation for text based Information retrieval and how it can be improved. Section 3 introduces the document model residuals and how we can compute them. Section 4 provides a small example of how to compute the document model residuals. Section 5 contains the analysis of residuals from a document collection. Finally, section 6 presents the work that is related to the content of this article.

2 Document model evaluation

A document model is a representation of a document for use in information retrieval. The most common document models are functions that take a document and statistics from the document collection, to compute weights representing the relevance of each term to the document in question.

There are three main classes of document models for text retrieval: vector space models, probabilistic models and language models. Each of the document models use the same linear function to compute document relevance scores:

$$s_d = \sum_{q \in Q} m_{\text{model}}(d, t, C) w_{q,t}$$

where s_d is the document relevance score, $m_{\text{model}}(d, t, C)$ is the weight of term t in document d computed using a document model and statistics from the document collection C, and $w_{q,t}$ is the weight of term t in query q.

The vector space document model [6] generally follows the TF-IDF (term frequency - inverse document frequency) form:

$$m_{\text{VSM}}(d, t, C) = f_{d,t} \log\left(\frac{N}{f_t}\right) \tag{1}$$

where $f_{d,t}$ is the frequency of term t in document d, f_t is the number of documents containing term t, and N is the number of documents.

A popular probabilistic document model is the BM25 model [3], which has the form:

$$m_{\text{BM25}}(d, t, C) = \frac{f_{d,t}(k_1 + 1)}{f_{d,t} + k_1(1 - b + b l_d/\bar{l})} \times \log\left(\frac{N - f_t + 0.5}{f_t + 0.5}\right) \tag{2}$$

where l_d is the length of document d, $\bar{l}$ is the average document length, and k_1 and b are parameters.

The language model document model function [8] has the form:

$$m_{\text{LM}}(d, t, C) = \log\left(\alpha P(t|d) + (1 - \alpha) P(t|C)\right)$$

where $P(t|d)$ is the probability of term t being sampled from document d, $P(t|C)$ is the probability of term t being sampled from the document collection C, and α is the smoothing parameter.

To evaluate the ranking produced from the document model, we must have a set of relevance judgements for the query. Relevance judgements are values that are assigned to each document for each query, reflecting the relevance of the associated document to the associated query. Relevance judgements may be binary (1 meaning relevant, 0 meaning irrelevant), or ordinal (0 meaning irrelevant, and 1, 2, 3, 4 and 5 being levels of relevance, where 5 implies the greatest relevance). We compare the document model relevance scores to the relevance judgements using an evaluation function. The evaluation function compares the ranking of documents (induced by the relevance scores) to the relevance judgements. A perfect score is awarded to document models that score documents of greater relevance over those of less relevance (or in the binary case, score all relevant documents greater than the set of irrelevant documents). A popular evaluation function for binary relevance judgements is average precision (AP) [4]. Average precision provides us with a score between 0 and 1 inclusive, where 1 is given for a perfectly ranked list.

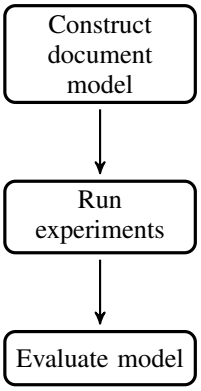

Figure 1: Current method of evaluating document models. There is no method of identifying if the document model can be improved, based on the outcome of the evaluation.

The process of document model evaluation in previous retrieval experiments has stopped here. Once the document model is evaluated, we obtain an evaluation score for the model, and can then compare it to other scores from other document models. This one way process is shown in Figure 1. There is currently no method of examining where the differences in document models lie, or where a model is performing poorly. In the next section, we will examine how we can assess document model deficiencies and their suitability for a given collection, using document model residuals.

3 Evaluation feedback using residuals

The construction of a document model is a regression problem, where we are mapping a set of document and query attributes to a real scalar value that reflects the relevance of the associated document to the associated query. Therefore, we are able to assess the goodness of fit of a document model by examining its residuals. Document model residuals show us how each element of the document model is affecting the accuracy of the document model. By examining the document model residuals, we obtain feedback on how the document model can be improved, and hence close the loop on Information retrieval evaluation (shown in Figure 2).

We saw in the previous section that many of the document models are linear functions; this linearity is one of the factors that leads to fast query times and so is preferred over non-linear models. The linear model can be given as:

$$\vec{s}_d = \vec{d}Q^T \quad \text{or} \quad \vec{s}_q = D\vec{q}^T$$

where $\vec{s}_d$ is the vector of relevance scores for document d predicted for all queries, and $\vec{s}_q$ is the vector of relevance scores for all documents predicted for queries q, $\vec{d}$ is the vector of document weights that are computed using the document model, D is the matrix containing vectors $\vec{d}$ as its rows, $\vec{q}$ is a vector of query weights, Q is a row matrix of query vectors $\vec{q}$, and Q^T is the transposed Q.

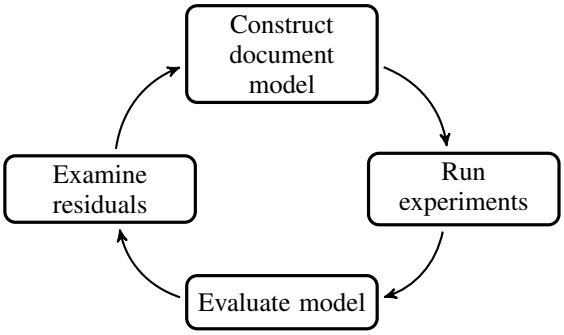

Figure 2: Suggested document model improvement cycle for information retrieval. Using the evaluation results, we are able to construct the document model residuals and examine where the model can be improved.

Using this linear form of retrieval, we can examine two forms of residuals: *relevance score residuals* and *document weight residuals*. Relevance score residuals are the difference between the ideal relevance scores and the relevance scores predicted by the document model. Document weight residuals are the difference between the document weights computed using the document model and the document weights that would lead to the ideal relevance scores.

3.1 Relevance score residuals

The best document model weights Δ for a given document-query set pair can be shown as:

$$\vec{s}_q = \Delta\vec{q}^T \tag{3}$$

where the document score vector $\vec{s}_q$ is the solution to:

$$\vec{s}_q = \arg\max_{\vec{\theta}_q} \left(\text{eval}\left(\vec{r}_q, \vec{\theta}_q\right) \right)$$

Our document model, used to compute the values in D are an attempt to achieve the ideal document model values in Δ, and are therefore an approximation to Δ. The difference between D and Δ is the set of document model residuals E. Therefore, we can write equation 3 as:

$$\begin{aligned} \vec{s}_q &= (D + E)\,\vec{q}^T \\ &= D\vec{q}^T + E\vec{q}^T \end{aligned} \tag{4}$$

The first term in 4 is the set of scores generated by our document model, while the second term is the set of scores that need to be added to our model's scores to provide the best document ranking:

$$\vec{s}_q = \vec{s}_{D,q} + \vec{s}_{E,q}$$

where $\vec{s}_{D,q}$ is the set of scores using our document model, being an approximation to $\vec{s}_q$, and $\vec{s}_{E,q}$ is the difference in $\vec{s}_q$ and $\vec{s}_{D,q}$, or the relevance score residuals. In order to identify the set of relevance score residuals, we should choose $\vec{s}_{E,q}$ to maximise the evaluation metric being used. We can show this as:

$$\vec{s}_{E,q} = \arg\max_{\vec{s}_{\Xi,q}} \left(\text{eval}\left(\vec{r}_q, \vec{s}_{D,q} + \vec{s}_{\Xi,q}\right)\right) \quad (5)$$

There are many solutions to this optimisation problem, since any vector that we add to $\vec{s}_{D,q}$ to give the correct ranking is a solution. Therefore, we add the extra constraint that we want the relevance score residuals that are a solution to equation 5, and cause the smallest deviation to the document model scores $\vec{s}_{D,q}$:

$$\vec{s}_{E,q} = \arg\max_{\vec{s}_{\Xi,q}} \left(\text{eval}\left(\vec{r}_q, \vec{s}_{D,q} + \vec{s}_{\Xi,q}\right) - \gamma \|\vec{s}_{\Xi,q}\|_2\right) \quad (6)$$

where γ is a positive constant that controls the residual minimisation, and $\|\cdot\|_2$ is the l_2 vector norm.

To solve the optimisation problem in equation 6, we need to obtain the gradient of the function being optimised. Unfortunately, the function contains $\text{eval}(\cdot,\cdot)$. If we examine the set of evaluation functions for information retrieval, we find that they are all disjoint. As the score vector transitions through the vector space, the evaluation score will not change until the ranks of the score vector elements change. Therefore, the evaluation function is a series split of levels, not a smooth function. This means that the gradient of the evaluation metrics are not continuous and hence we are unable to solve the maximisation by examining the evaluation function gradient.

Rather than propose a complex method to solve the optimisation problem in equation 6, we have chosen to add a further constraint to make the problem simple to solve. We add the constraint that the elements of $\vec{s}_{\Xi,q}$ must be non-negative, giving us the optimisation problem:

$$\vec{s}_{E,q} = \arg\max_{\vec{s}_{\Xi,q}} \left(\text{eval}\left(\vec{r}_q, \vec{s}_{D,q} + \vec{s}_{\Xi,q}\right) - \gamma \|\vec{s}_{\Xi,q}\|_2\right)$$
$$\text{such that } s_{\Xi,q,d} \geq 0$$

where $s_{\Xi,q,d}$ are the elements of $\vec{s}_{\Xi,q}$.

When examining the case of binary relevance judgements and γ is small, we find that the solution to this problem is the vector that increases all relevant document scores to be at least slightly greater than the score of each irrelevant document. The algorithm to compute this vector is provided in Algorithm 1.

Algorithm 1 Compute relevance score residuals $\vec{s}_{E,q}$ where ϵ is a small positive real value.

1: $\vec{s}_q = \vec{s}_{D,q}$
2: Find in $\vec{s}_{D,q}$, the greatest score assigned to an irrelevant document $s_{i,q}$.
3: **for** each relevant document score $s_{r,q}$ in $\vec{s}_q$ **do**
4: **if** $s_{r,q} \leq s_{i,q}$ **then**
5: $s_{r,q} = s_{i,q} + \epsilon$
6: **end if**
7: **end for**
8: $\vec{s}_{E,q} = \vec{s}_q - \vec{s}_{D,q}$

3.2 Document model residuals

The document model residuals are the difference in the document weights computed using the document model, and the document weights of the ideal document model. We can extend on the optimisation used to compute the relevance score residuals in equation 6, to compute the document model residuals. We do this by replacing the relevance scores with the document model weights used to compute the relevance scores:

$$E = \arg\max_{\Xi} \left(\text{eval}\left(\vec{r}_q, D\vec{q}^T + \Xi\vec{q}^T\right) - \gamma \|\Xi\vec{q}^T\|_2\right) \quad (7)$$

Again, this is a complex optimisation since the $\text{eval}(\cdot,\cdot)$ is not likely to have a continuous gradient. Instead of solving the problem in equation 7, we have broken the problem into two stages, making the optimisation problem easier to compute. The first stage involves computing the relevance score residuals (examined in section 3.1), the second stage involves computing the document model residuals from the relevance score residuals.

The ideal document model scores are those that provide the best precision for the document-query set. Computing the ideal document model weights, given the ideal document scores, is a regression problem:

$$\vec{s}_\delta = \vec{\delta} Q^T$$

where $\vec{\delta}$ is the vector of ideal document weights for document d, Q is the matrix of query vectors, and $\vec{s}_\delta$ is the set of ideal document scores for document d. The document scores can be written as:

$$\begin{aligned}\vec{s}_\delta &= \vec{d} Q^T + \vec{e} Q^T \\ &= \vec{s}_{d,Q} + \vec{s}_{e,Q}\end{aligned}$$

where $\vec{d}$ is the vector of weights computed using our document model, $\vec{s}_{d,Q}$ is the set of scores of document d, $\vec{e}$ is the vector of document model residuals, and $\vec{s}_{e,Q}$ is the difference between the ideal document model scores and our document model scores.

We know the values in Q, $\vec{d}$, $\vec{s}_\delta$ and $\vec{s}_{d,Q}$. From these, we can easily calculate $\vec{s}_{e,Q}$ and so can compute the vector of document model residuals $\vec{e}$ using linear regression:

$$\vec{e} = \arg\min_{\vec{\eta}} \|\vec{s}_{e,Q} - \vec{\eta} Q^T\|_2$$

But, note that since we usually have a small number of queries, and a large vector space, the regression is computed with zero degrees of freedom, hence the vector $\vec{e}$ will overfit the document-query set, and not generalise well to other queries. To increase the generalisation of $\vec{e}$, we regularise the regression, and compute the ideal document weights using ridge regression [2]:

$$\vec{e} = \arg\min_{\vec{\eta}} \left(\|\vec{s}_{e,Q} - \vec{\eta} Q^T\|_2 - \lambda \|\vec{\eta}\|_2\right) \quad (8)$$

where λ is a real positive value that controls the level of regularisation.

4 Residual computation example

In this example, we have used a document model to compute the term weights for each document and stored the weights in the matrix D, where each row of D represents the term weights for a specific document in the document collection. We have also constructed the query matrix Q containing the weights associated to each query. The query weights are binary, showing the absence or presence of a term:

$$D = \begin{bmatrix} 1 & 1 & 2 & 2 & 0 \\ 1 & 2 & 5 & 2 & 0 \\ 0 & 1 & 1 & 2 & 1 \\ 2 & 0 & 4 & 1 & 2 \\ 4 & 0 & 5 & 1 & 1 \end{bmatrix} \quad Q = \begin{bmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{bmatrix}$$

We compute the predicted relevance scores as the inner product of each document vector with each query vector, given as $\hat{S} = DQ^T$. The contents of $\hat{S}$ is shown below with the relevance judgement matrix R:

$$\hat{S} = \begin{bmatrix} 3 & 3 & 5 & 3 \\ 4 & 3 & 8 & 6 \\ 4 & 2 & 3 & 1 \\ 3 & 3 & 7 & 6 \\ 2 & 5 & 10 & 9 \end{bmatrix} \quad R = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

The mean average precision (MAP) of this document model on this document-query set is computed using $\hat{S}$ and R as 0.675.

The goal of each document model is to compute a relevance score for each document, so that the documents judged relevant are scored greater than those judged irrelevant. To compute the relevance score residuals, we must first compute the ideal set of relevance scores, that have little variation on the predicted relevance scores $\hat{S}$. We compute these scores using the first seven lines of Algorithm 1. This gives us the ideal relevance score matrix:

$$S = \begin{bmatrix} 3 & 3 & 5 & 3 \\ 4 & 3 & 10+\epsilon & 9+\epsilon \\ 4 & 2 & 3 & 9+\epsilon \\ 3 & 3+\epsilon & 7 & 6 \\ 2 & 5 & 10 & 9 \end{bmatrix}$$

We compute the relevance score residuals S_E as the difference between the ideal relevance scores S and the relevance scores predicted using the document model $\hat{S}$ (using the last line of Algorithm 1):

$$S_E = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 2+\epsilon & 3+\epsilon \\ 0 & 0 & 0 & 8+\epsilon \\ 0 & \epsilon & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Using the relevance score residual matrix and the query matrix, we can compute the document model residuals E using ridge regression. If we use a small value of λ (in this case $\lambda = 10^{-10}$), we obtain the set of document weight residuals that provide perfect precision:

$$E = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0.5 & 2+10^{-6} & -1 & 0.5 \\ 8+10^{-6} & 4 & 0 & -8+10^{-6} & 4 \\ 10^{-6} & 0 & -10^{-6} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

By adding these residuals to our document weights in D and applying the queries, we obtain a mean average precision of 1. Unfortunately, by using a small value of λ, we usually over fit the document-query set, meaning that the residuals are a reflection of the data rather than the document model. To obtain more informative residuals, we should use a larger value of λ. The larger the value of lambda, the greater the generalisability of the residuals to other query sets, but we obtain a smaller increase in precision. For this example, we have chosen to examine the document weight residuals for $\lambda = 0.1$, and obtain:

$$E = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0.885 & 0.372 & 1.910 & -0.782 & 0.372 \\ 5.140 & 2.619 & 1.533 & -5.501 & 2.619 \\ 10^{-6} & 0 & -10^{-6} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

By adding this set of document weight residuals to the document weight matrix D and applying the queries, we obtain a mean average precision of 0.854.

By examining the document weight residuals, we see that the largest residuals exist on the third row, which is associated to the third document. This may indicate that the current document model is not sufficient for the third document. We can also see that the largest residuals exist in the first and fourth columns (associated to the first and fourth terms). This is an indicator that the term weighting within the document model may need adjusting to suit these terms.

5 Examining document model residuals

In the previous sections, we provided a framework for computing document model residuals. In this section we will compute the residuals of the raw document model, where the weights of the model are the term frequencies, and the BM25 document model, where the weights of the document model are the BM25 term weights. After computing the residuals, we will examine the correlation between the residuals and statistics of the document collection to demonstrate how we can adjust the document models to provide a better fit to the document collection. We will perform this analysis using the CRAN document collection[2], which contains 1398 documents, but most importantly 225 long queries, and manual relevance judgements for all documents on each query. Note that residuals can only be found on terms that are within queries and documents that have relevance judgements. Therefore this set will provide us with many residuals.

5.1 Raw term frequency residuals

First, we will examine the raw term frequencies and demonstrate how a well known form of term weight can

[2] `ftp://ftp.cs.cornell.edu/pub/smart/`

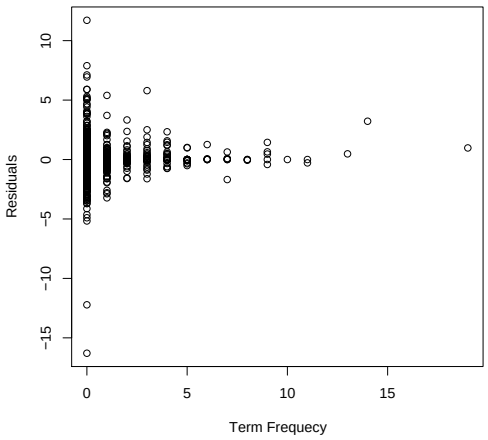

Figure 3: Residuals of the term frequencies for all documents compared to the term frequency values.

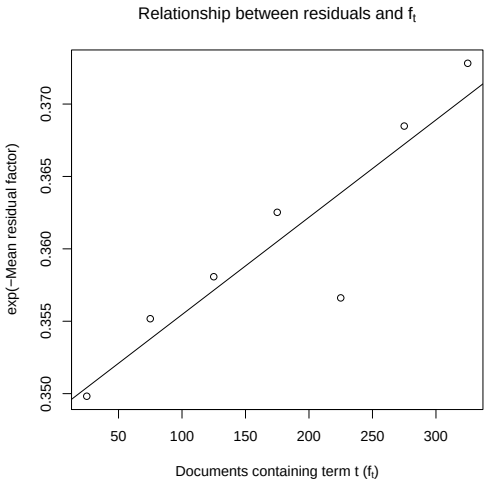

Figure 4: The correlation between f_t (the number of documents the term appears in) and the exponential of the negative mean residual factor associated to the f_t value. The line shows the least squares fit to the points.

be observed using the residuals. To compute the residuals, we construct the matrix D containing the raw term frequencies $f_{d,t}$ and the matrix Q containing binary values; 1 if the associated term appeared in the query, and 0 if the term did not appear in the query. We then obtained the set of relevance score residuals, with the constraint that the residuals were not negative, using Algorithm 1. Once we obtained the set of relevance score residuals, we computed the document model residuals using ridge regression from equation 8 with $\lambda = 1$. The plot of the residuals of each term frequency is shown in Figure 3.

The first observation of Figure 3 is that the residuals are almost symmetric about zero for each term frequency, and the that variance of the residuals decreases as the term frequency value increases. This shows that we need to modify the values of small term frequencies to achieve δ, the ideal model for the collection, but larger term frequencies need little modification. We can also see that there are many large residuals associated to the term frequency zero; this reflects the importance of having non-zero weights associated to term frequencies of zero.

We will now examine the correlation of the document model residuals to the term document count f_t. To do this, we examined the mean residual for values of f_t between 0 and 350, where the mean is computed for f_t using bins of width 50. We found no obvious relationship between the document model residuals and f_t. Therefore, we also examined the document model residual factor, computed as $(e_{d,t} + f_{d,t})/f_{d,t}$. A plot comparing the mean document model residual factor and f_t is shown in Figure 4.

Using this plot in Figure 4, we computed the linear relationship:

$$\exp\left(-\frac{e_{d,t} + f_{d,t}}{f_{d,t}}\right) = 0.35 + 6.72 \times 10^{-5} f_t$$

which suggests the improved document model:

$$f_{d,t} \log\left(\frac{1}{a + b f_t}\right) \tag{9}$$

where a and b are parameters determined by the document collection; in this case $a = 0.35$ and $b = 6.72 \times 10^{-5}$. This form of document model is very similar to the VSM document model (shown in equation 1) which provides an improvement over the raw term frequency model. Therefore, by examining the document model residuals, we have shown the importance of weighting by the log of the inverse term weight f_t.

To examine the effect of the modified term frequency document model, we constructed the matrix D containing the values computed using equation 9, and compared the increase in each query's AP to the AP of the raw term frequency model. A histogram showing the distribution of the change in AP for each query is provided in Figure 5.

We can see from this histogram that an increase in AP was provided to a large set of queries (172 queries), and a decrease in AP was provided to a small set of queries (45 queries). The paired Wilcoxon signed rank test on the AP produced from the raw term frequencies compared to the modified term frequencies provided a p value of less that 10^{-16}, showing that the modification produced by observing the residuals provided a significant increase in AP. The mean AP was increased from 0.2253 to 0.2316.

5.2 BM25 residuals

We will now examine the BM25 document model and try to identify if there is any correlation of the document weight residuals to the document collection statistics. We construct the matrix D using the BM25 document weights from equation 2 and the matrix Q containing binary values reflecting the appearance of the associated term in the query.

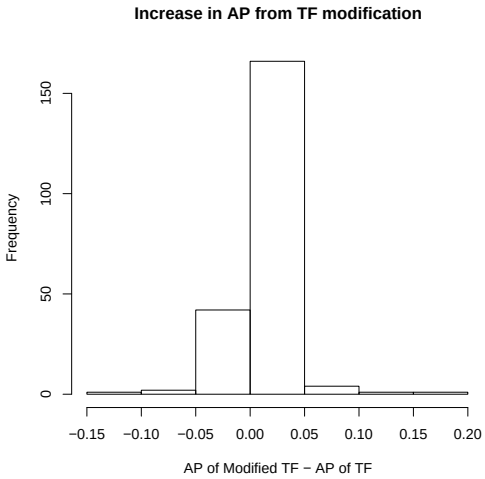

Figure 5: Distribution of the increase in AP after modifying the raw term frequencies with the results from the residual analysis.

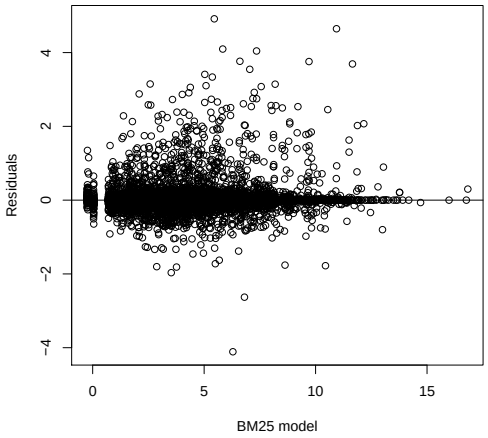

Figure 6: Residuals of the BM25 document weights for all documents compared to the model weights.

We again obtained the set of relevance score residuals using Algorithm 1. We computed the document model residuals using ridge regression from equation 8 with $\lambda = 1$. The plot of the residuals of each BM25 term weight is shown in Figure 6.

We can see from Figure 6 that there is small variance in the residuals for small and large BM25 weights, but the variation increases near the region where the BM25 weight is 5. This shows that the mid-range BM25 weights are not as accurate at the small and large values, hence improvements can be made. We can see that there is also a set of residuals where the weight is 0 (implying the term frequency is 0). The residuals at zero are not as large as found in the term frequency plot, but still present, implying that well calculated non-zero weights for zero valued

Figure 7: The correlation between the BM25 term weights and the square root of the mean residual of the associated term weight. The line shows the least squares fit to the points.

term frequencies would improve the accuracy of the document model.

We will now examine the average residual error compared to the BM25 weight. The BM25 weight is a continuous value, therefore, to compute the average residual, we gathered all BM25 weights into 0.2 width bins and computed the average residual associated to each weight in the bin. The correlation observed is shown in Figure 7. The line fit to the plot in Figure 7 depicts the relationship:

$$\sqrt{e_{d,t}} = 0.026 m_{\text{BM25}}(d,t,C) + 0.038$$

which suggests the improved document model:

$$m_{\text{BM25}}(d,t,C) + (a + b m_{\text{BM25}}(d,t,C))^2 \qquad (10)$$

where a and b are parameters determined by the document collection; in this case $a = 0.038$ and $b = 0.026$. This residual analysis has suggested that we should use a quadratic form for our document model. To examine the improvement given by this model, we constructed the matrix D containing the values from equation 10 and compared the increase in each query's AP to the AP of BM25. A histogram showing the distribution of the change in AP for each query is provided in Figure 8.

We can see from this histogram that an increase in AP was provided to a set of queries (51 queries), and a decrease in AP was provided to a smaller set of queries (32 queries). The paired Wilcoxon signed rank test on the AP produced from the BM25 weights compared to the modified BM25 weights provided a p value of 0.0240 showing that the modification produced by observing the residuals provided a significant increase in AP. Given all of this, we found that the mean AP decreased from 0.3620 to 0.3617. By examining the histogram in Figure 8 we can see that the decrease would be due to the one query that caused the small bar at -0.06 to -0.05.

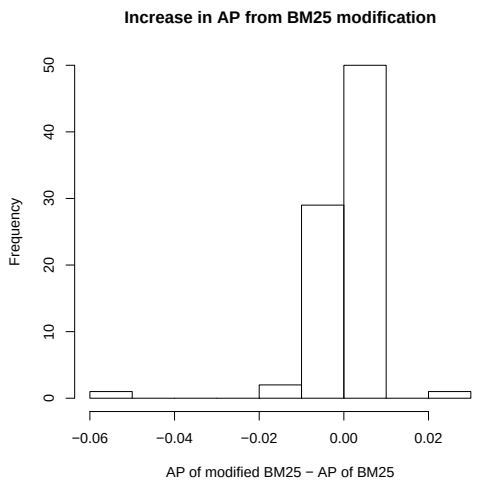

Figure 8: Distribution of the increase in AP after modifying BM25 with the results from the residual analysis.

Note that even though we have derived new document models, we do not suggest that these should be used in practice. These models were obtained by examining the residuals on a single document set using a small number of queries. Further analysis such as cross validation should be performed in order to better examine the effects of modified document models and to determine the value of λ that should be used. We included this analysis simply to demonstrate how the residuals could be used in practice.

6 Related Work

There are similarities between the residual computation we have proposed and the work done by the learning to rank community [5, 1, 7]. Learning to rank is the process of using machine learning methods to a construct document models for specific documents-query sets. When using learning to rank methods, we obtain a document model that has high precision, but the models are usually complex and so we are unsure why the model provides high precision.

The work we have proposed is not for automatically constructing high precision document models, but for examining where our existing document models work well and where they are deficient. By using the methods presented in this article, we will gain a better understanding of what is required to obtain high precision for text document sets from any domain.

7 Conclusion

Information retrieval evaluation is needed in order to determine the document models that are most suitable for retrieval tasks. The existing evaluation process provides a score to each document model, allowing us to compare models, but unfortunately, there is no method of determining where the document models are performing well and where they are deficient.

In this article, we presented a framework for computing document model residuals. By examining the residuals of a document model, we can discover the difference between the document model weights and the weights of the ideal document model. We can also use the document model residuals to examine correlation between document collection statistics, and hence improve the model.

We included an analysis of the document model residuals for the raw term frequency model and the BM25 document model. We showed that we were able to find correlation between the residuals and statistics, such as the term document count and the within document term frequency.

By computing and analysing the document model residuals, we are able to close the gap in the evaluation cycle, and reduce the black box evaluation that is currently performed. This will allow us to develop accurate domain specific document models more intelligently.

References

[1] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton and Greg Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*, ICML '05, pages 89–96, New York, NY, USA, 2005. ACM.

[2] Trevor Hastie, Robert Tibshirani and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Series in Statistics. Springer, 2nd edition, 2001.

[3] K. Sparck Jones, S. Walker and S. E. Robertson. A probabilistic model of information retrieval: development and comparative experiments, part 2. *Information Processing and Management*, Volume 36, Number 6, pages 809—840, 2000.

[4] Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze. *Introduction to Information Retrieval*, Chapter 8: Evaluation in information retrieval. Cambridge University Press, 2008.

[5] Ramesh Nallapati. Discriminative models for information retrieval. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '04, pages 64–71, New York, NY, USA, 2004. ACM.

[6] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, Volume 24, Number 5, pages 513 – 523, 1988.

[7] Jun Xu and Hang Li. Adarank: a boosting algorithm for information retrieval. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 391–398, New York, NY, USA, 2007. ACM.

[8] Chengxiang Zhai and John Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 334–342, New York, NY, USA, 2001. ACM Press.

[9] Justin Zobel and Alistair Moffat. Exploring the similarity space. *ACM SIGIR Forum*, Volume 32, Number 1, pages 18–34, Spring 1998.