Reducing the Risk of Query Expansion via Robust Constrained Optimization

Kevyn Collins-Thompson
Microsoft Research
1 Microsoft Way
Redmond, WA 98052-6399 U.S.A.
kevynct@microsoft.com

ABSTRACT

We introduce a new theoretical derivation, evaluation methods, and extensive empirical analysis for an automatic query expansion framework in which model estimation is cast as a robust constrained optimization problem. This framework provides a powerful method for modeling and solving complex expansion problems, by allowing multiple sources of domain knowledge or evidence to be encoded as simultaneous optimization constraints. Our robust optimization approach provides a clean theoretical way to model not only expansion benefit, but also expansion risk, by optimizing over *uncertainty sets* for the data. In addition, we introduce *risk-reward curves* to visualize expansion algorithm performance and analyze parameter sensitivity. We show that a robust approach significantly reduces the number and magnitude of expansion failures for a strong baseline algorithm, with no loss in average gain. Our approach is implemented as a highly efficient post-processing step that assumes little about the baseline expansion method used as input, making it easy to apply to existing expansion methods. We provide analysis showing that this approach is a natural and effective way to do selective expansion, automatically reducing or avoiding expansion in risky scenarios, and successfully attenuating noise in poor baseline methods.

Categories and Subject Descriptors: H.3.3 [**Information Retrieval**]: Retrieval Models
General Terms: Algorithms, Experimentation
Keywords: Query expansion, convex optimization

1 Introduction

Despite decades of research on automatic query expansion [12], even state-of-the-art methods suffer from drawbacks that have limited their deployment in real-world scenarios. One example of this is their failure to account for the tradeoff between risk and reward: current techniques are optimized to perform well on average, but are unstable and have high variance across queries [10]. They are ineffective at avoiding risk by operating *selectively*, reducing or avoiding expansion when it is likely to dramatically decrease performance. In addition, Web search engines must support an increasingly complex decision environment in which potential enhancements to a query may be influenced and constrained by multiple factors including personalization [26], implicit and explicit relevance information, and computation budgets. Expansion terms are usually selected individually, when the correct approach should be to optimize over the entire solution as a set. This is particularly problematic for Web search, where the set of related words is forced to be very small for speed and reliability reasons.

Existing closed-form term-weighting formulas for query expansion find such scenarios difficult or impossible to handle, and cannot easily balance risk and reward. For automatic expansion algorithms to be widely used, is important to find techniques that maintain the good average performance of existing methods, but which are more general and more reliable.

Convex optimization methods [3] are a special class of optimization technique that includes well-known algorithms such as least-squares, but with more general capabilities, so that they can handle a much wider set of problems. They provide an attractive starting point for several reasons. Their ability to integrate different criteria for expansion, such as risk and reward, as optimization constraints and objectives allows us to express an extremely rich set of expansion scenarios in simple form. Convex programs also have unique, globally optimal solutions and reliable, highly efficient solution methods. For example, the quadratic programs we develop below can now be solved in just tens of milliseconds for a few hundred variables. Finally, many widely-used functions used in IR tasks, such as vector dot products and KL-divergence, are convex functions, making it possible to cast realistic IR problems as convex programs.

We contribute a new general formulation of query expansion as a convex *robust optimization problem* that produces more conservative solutions by optimizing with respect to *uncertainty sets* defined around the observed data. Because our goal is to mitigate the risk-reward tradeoff of expansion, we introduce risk-reward curves as an evaluation tool. We make a detailed study of algorithm constraints, and demonstrate that ignoring risk leads to less stable expansion algorithms. Our approach significantly reduces the number and magnitude of expansion failures of a strong baseline method with no loss in average gain, while successfully attenuating noise when the baseline is poor quality.

CIKM'09, November 2–6, 2009, Hong Kong, China.

2 Theoretical model

Our goal in this section is to show how query expansion can be cast as a robust convex optimization problem. Instead of deriving a term weighting formula in closed form, we will define a set of *objectives* and *constraints* that good expansion models would be expected to satisfy. We then give the resulting convex program to a *solver*, which either finds an optimal point $x^\star$ satisfying the constraints, or determines that no solution is *feasible*, i.e. can satisfy all constraints.

We define a *query expansion* for a query Q of length K as a set of weights $x = (x_1, \ldots, x_n)$ over terms in a vocabulary $\mathcal{V}$, with $x_i \in [0, 1]$. Each x_i represents a weight for term $w_i \in \mathcal{V}$. If used with a rounding threshold, x_i can also be thought of as a soft decision to include or exclude the term from the expansion[1]. We assume our observations are the initial query Q, and a set of resulting top-ranked documents for the query $\mathcal{D}$.

For query expansion, the key idea is that we want to balance two criteria: *reward* and *risk*. The reward criterion for a feasible expansion x, which we denote $R(x)$, reflects the selection of 'good' individual expansion terms, while the risk criterion, denoted $V(x)$ penalizes choices for which there is greater uncertainty, both for individual terms and the entire expansion set. We now develop these in more detail.

2.1 Reward objectives

In general, obtaining an appropriate objective function for query expansion would seem to be very difficult, especially since we want general-purpose methods that can be easily applied to existing algorithms. We want to make few assumptions about the nature of the retrieval model.

Our solution is to view query expansion in two phases: the first step obtains initial candidate weights $p = \Phi(Q, \mathcal{D})$ using a baseline expansion method Φ, that we treat as a black box. The second step is a constrained optimization phase applied to the initial candidates p to obtain the final expansion solution $x^\star$. In that sense, our approach can be seen as a general-purpose wrapper method. Note that we do *not* assume the baseline method is 'reasonable': in fact, we show in Section 4.6 that our approach can tolerate even the most ill-behaved baselines.

With this view, choosing a reward objective becomes much easier: we want to bias the solution toward those words with the highest p_i values. One simple, effective function is the *expected relevance* of a solution x: the weighted sum $R(x) = p \cdot x = \sum_k p_k x_k$. Other choices for $R(x)$ are certainly possible, although we do not explore them here. For example, if p and x represent probability distributions over terms terms, then we could choose KL-divergence $R(x) = KL(p||x)$ as an objective since it is convex.

For Indri's language model-based expansion, we have Relevance Model estimates $p(w|R)$ over the highest-ranking k documents, where the symbols R and N represent relevance and non-relevance respectively. We can also estimate a non-relevance model $p(w|N)$ to approximate non-relevant documents using either the collection, or the bottom-ranked k documents from the initial query Q. To set p_i, we first compute $p(R \mid w)$ for each word w via Bayes Theorem, $p(R|w) = \frac{p(w|R)}{p(w|R)+p(w|N)}$ assuming $p(R) = p(N) = 1/2$. Then, using $p(R|Q)$ and $p(R|\bar{Q})$ to denote the probability that any query word or non-query word respectively should be included in the expansion, the expected value is then

$$p_i = \begin{cases} p(R|Q) + (1 - p(R|Q)) \cdot p(R|w_i) & w_i \in Q \\ p(R|\bar{Q}) \cdot p(R|w_i) & w_i \notin Q. \end{cases} \quad (1)$$

We choose $p(R|Q) = 0.75$ and $p(R|\bar{Q}) = 0.5$.

2.2 Risk objectives

If reward were our only objective, maximizing the objective $R(x)$ would simply correspond to preferring the expansion terms with highest p_i values assigned by the baseline expansion algorithm, given whatever constraints or other conditions, such as sparsity, that were in force. These baseline p_i weights, however, are inherently uncertain, and the final expansion solution may vary considerably for different hypotheses about the 'true' p_i values. To account for this uncertainty, we invoke *robust optimization* methods. Robust methods model uncertainty in p by defining an *uncertainty set* $\mathcal{U}$ around p, and then applying a minimax approach – that is, minimizing the worst-case solution over all possible expansions realized by p varying in its uncertainty set $\mathcal{U}$. The result is a more conservative algorithm that avoids relying on data in directions of higher uncertainty.

2.2.1 Uncertainty in expansion term weights

We treat p as a random variable varying within an *uncertainty set* $\mathcal{U}$ distributed with mean p and covariance Σ. We define $\mathcal{U}_\kappa$ to be an ellipsoidal region around p

$$\mathcal{U}_\kappa = \{u \in \mathbb{R}^n | (u - p)^T \Sigma^{-1} (u - p) \leq \kappa^2\}. \quad (2)$$

We set the covariance matrix entries Σ_{ij} by choosing a term similarity measure $\sigma(w_i, w_j)$ and define $\Sigma_{ij} = d(w_i, w_j) = \exp(-\eta \cdot \sigma(w_i, w_j))$, where the constant η is a normalization factor that depends on the measure chosen for $\sigma(w_i, w_j)$. In this study, $\sigma(w_i, w_j)$ was defined using a word similarity measure based on query perturbations described in [6]. In practice, however, we have also used simpler methods for $\sigma(w_i, w_j)$, such as the Jaccard similarity[2] over the 2-by-2 contingency table of term frequencies for w_i and w_j in the top-ranked documents, with only small reductions in effectiveness.

The adjustable parameter κ represents our tolerance of risk, with larger κ indicating a higher aversion to risk. When $\kappa = 0$, the uncertainty set $\mathcal{U}_0$ is a single point, p, and we have our original reward-only problem. If $\Sigma = \boldsymbol{I}$, $\mathcal{U}_\kappa$ is the ellipsoid of largest volume inside box $B = \{u \,|\, |u_i - p_i| \leq \kappa\}$. The key result we now use is the following theorem from Ben-Tal & Nemirovski [2].

THEOREM 1. *The robust counterpart of an uncertain linear program with general ellipsoidal uncertainty can be converted to a conic quadratic program.*

As an example, applying this theorem to the very simple reward-only linear program

$$\begin{aligned} \text{minimize} \quad & -p^T x & (3) \\ \text{subject to} \quad & 0 \leq x_i \leq 1 & (4) \end{aligned}$$

[1] Useful for search engines without term weighting operators.

[2] The Jaccard similarity is defined as $\sigma_J(w_i, w_j) = M_{11}/(M_{01} + M_{10} + M_{11})$ where the M_{jk} are the cells of the contigency table containing total counts in the top-ranked documents, and j, k are the binary variables indicating presence or absence of w_i and w_j respectively in a document.

results in the robust quadratic program version

$$\text{minimize} \quad -p^T x + \frac{\kappa}{2} x^T \Sigma x \tag{5}$$

$$\text{subject to} \quad 0 \leq x_i \leq 1 \tag{6}$$

when p varies within the uncertainty set $\mathcal{U}_\kappa$ as defined above. We thus have our risk objective $V(x) = \frac{\kappa}{2} x^T \Sigma x$. Combining risk and reward gives the bi-objective function $U(x) = -p^T x + \frac{\kappa}{2} x^T \Sigma x$.

2.2.2 Uncertainty in the covariance matrix Σ

We can make the solution even more conservative by defining an uncertainty set for Σ itself. We do this using a simple diagonal matrix W, with W_{ii} reflecting uncertainty in the estimate of Σ_{ii}. To set W_{ii}, we introduce the idea of *term centrality*. Previous work [9][29] has shown that terms are more reliable for expansion if they are related to multiple query terms. We thus define W_{ii} in terms of the vector d_i of all similarities of w_i with all query terms. This gives $W_{ii} = \|d_i\|_2^2 = \sum_{w_q \in Q} d^2(w_i, w_q)$. It can be shown [15] that the effect of this regularization is a modified covariance matrix $\Sigma_\gamma = \Sigma + \gamma^{-1} W$, where γ controls the relative influence of the diagonal W. Our joint reward and risk objective becomes

$$U(x) = -p^T x + \frac{\kappa}{2} x^T (\Sigma + \gamma^{-1} W) x. \tag{7}$$

We discuss the settings for κ and γ in Sec. 4.

2.3 Constraints for query expansion

We now add domain-specific constraints for query expansion: *aspect balance*, *aspect coverage*, and *query support*.

Aspect balance. Intuitively, we want an expansion that does not contain words that are all related to just one part of a query. Instead, there should be a balance among its component aspects. We assume a simplistic model where each query term represents a different aspect of the user's information need. We represent each query term q_k as a vector $\phi_k(w_i) = \Sigma_{ik}$ of its similarities to all words in $\mathcal{V}$. We then form a matrix A with K rows $A_k = \phi_k$. One way to express a 'balanced' solution x with respect to the aspect vectors ϕ_k is to require that the mean of the projection Ax be within a tolerance ζ_μ of the centroid $\mu = 1/K \sum \phi_k$, i.e. $Ax - \mu \preceq \zeta_\mu$ where $\preceq$ indicates component-wise comparison.

Query support. We also want the initial query terms q_k to have high weight in the optimal solution. We express this mathematically with simple box constraints, requiring x_i to lie above a threshold threshold $_i$ for $x_i \in Q$ as well as below the upper limit u_i, which is 1 for all terms. Term-specific values for l_i may also be desirable to reflect the rarity or ambiguity of individual query terms. We note that we could also implement query support using KL-divergence $KL(\theta_Q || x)$ in a probabilistic model.

The role of query support constraints differs from interpolation using α in model-based feedback since constraining feedback solutions to have high query support doesn't preclude significant expansion term support if there are many strongly related terms. However, as Section 4 shows, our risk framework is effective at finding expansion models that are closer to the ideal in which the 'right' amount of emphasis on the initial query terms is determined automatically. Our ultimate goal is to make model interpolation unnecessary, but we continue using it here to study the potential for further improvements.

$$\text{minimize} \quad -p^T x + \frac{\kappa}{2} x^T \Sigma_\gamma x + \lambda y \quad \textit{Reward \& risk} \tag{8}$$

$$\text{subject to} \quad Ax - \mu \preceq \zeta_\mu \quad \textit{Aspect balance} \tag{9}$$

$$g_i{}^T x \geq \zeta_i, \quad w_i \in Q \quad \textit{Aspect coverage} \tag{10}$$

$$l_i \leq x_i \leq u_i, \quad w_i \in Q \quad \textit{Query support} \tag{11}$$

$$w^T x \leq y \quad \textit{Budget/sparsity constraint} \tag{12}$$

$$0 \preceq x \preceq 1 \quad \textit{Label consistency} \tag{13}$$

Figure 1: The basic constrained quadratic program REXP used for query expansion.

Aspect coverage. This constraint is useful for retrieval oriented toward recall: it controls the absolute number of related words that are acceptable per query term. Similar to aspect balance, we denote the set of distances to neighboring words of query term q_k by the vector $g_k = \phi_k$. The projection $g_k{}^T x$ gives us the aspect coverage, or how well the words selected by the solution x 'cover' term q_k. The more expansion terms near q_k that are given higher weights, the larger this value becomes. We want the aspect coverage for each of the vectors g_k to exceed a threshold ζ_k, resulting in the constraint $g_k{}^T x \geq \zeta_k$, for all query terms q_k.

Putting together reward and risk objectives with the above constraints we obtain the final quadratic program, which we call REXP and is shown in Figure 1. To show the flexibility of our framework we give two useful extensions: setting budget constraints and finding top-k expansions.

2.4 Budget constraints

Since every expansion term with non-zero weight adds runtime cost for the search engine, we may prefer *sparse* solutions that minimize the number of expansion terms (while still respecting program constraints). More generally, some terms, such as those with very high frequency, may have a higher computation cost than rarer terms due to time required to load inverted lists from disk. All else being equal, we prefer solutions with lower probable computation cost. We handle such scenarios by introducing a weighted ℓ_1-norm penalty constraint with weight vector w. Non-zero entries for x_i will be discouraged when the corresponding w_i value is large. Since $x \geq 0$ this leads to the budget constraint $w^T x \leq y$. We can either set y to a fixed upper bound, or we can add it as a 'soft' constraint in the updated objective $U(x) = R(x) + \kappa V(x) + \lambda y$ where λ controls the relative importance of sparsity against risk and reward objectives. Setting $w = \mathbf{1}$ gives a standard ℓ_1-norm solution. We can penalize common terms by setting $w_i \propto f(t_i)$, where $f(t)$ is some function of term frequency of t in the index. If sparsity is critical, with more computation time we can use reweighted ℓ_1-norm minimization [13] to improve the initial ℓ_1 solution.

2.5 Producing top-k candidate lists

In some applications, we may wish to find a set of the best k alternatives, either for presenting alternative expansions to a user, or as a measure of confidence in the optimal expansion: if the sub-optimal expansion scores are far from the optimal we have more confidence in our estimate.

To do this, given an optimal solution $x^\star$, we gather a set of near-optimal points x^η by varying a rounding threshold η from 0 to 1 and setting entries with $x_i^\eta \leq \eta$ to zero. We

then calculate the objective function $U(x^\eta)$ for each of these alternate expansion solutions and sort by descending $U(x^\eta)$, selecting the top k expansions from this list[3].

3 Related work

Our work was initially inspired by the idea that query expansion can be viewed as a type of *portfolio allocation problem under uncertainty*, in which a set of financial securities must be selected that maximizes the expected return of the portfolio but also reduces risk by diversifying among different industry sectors [18]. Typically, there is the option of buying a safe 'risk-free' asset whose return is fixed and known in advance. Applied to IR, the user's query (in theory) represents the 'risk-free asset', and we can place bets on expansion terms[4]. IR has different task-specific constraints and estimating a 'rate of return' for the user's query may be problematic. Robust portfolio allocation methods are well-developed in the computational finance community but we have not seen much work in IR fully exploit this connection.

Our optimization constraints and objectives bring together several previous studies on the nature and causes of query drift [21][14]. Our constraint for aspect balance is based on observations from the 2003 RIA Workshop [4]. The empirically-derived Local Context Analysis (LCA) [29] includes a weighting factor preferring expansion terms that co-occur with multiple query terms. This corresponds to the term centrality criterion in our model. Downweighting the contribution of correlated terms has been shown to improve results [21], and our correlation matrix Σ plays a similar role. Our query support constraint keeps the expansion model 'close' to the query model, a condition that has been shown to be effective in other approaches [28][25]. The downside risk of query expansion has been noted for decades [23]: recently this problem has started to get some attention in evaluations [10][20][1] and we focus extensively on it here.

Optimization methods have been used implicitly and explicitly for IR problems. Implicit use has been via the use of machine learning techniques such as Support Vector Machines (SVM) Cao *et al.* [5] used an SVM to find good individual expansion terms but did not optimize over the expansion terms as a set, instead picking terms using a threshold. Explicitly, *unconstrained* optimization has been used for query expansion, typically using specialized code for a specific objective. Tao & Zhai [25] used EM with a non-convex, unconstrained likelihood objective to find a locally optimal expansion model, regularized using a prior that preferred models close to the initial query. Related work in smoothing [19] used gradient descent to smooth language models by minimizing a graph-based quadratic objective. To our knowledge, the use of robust optimization for IR-related problems has been limited to text classification [15].

In previous work [7] we introduced an early version of a Markowitz-type optimization framework for more reliable query expansion based on portfolio theory. Our work here greatly extends that initial study by providing the following contributions:

- A novel theoretical derivation in terms of *robust optimization* that allows us to define explicit uncertainty sets around variables of interest, including both individual term relevance scores and the term covariance matrix.
- New evaluation methods, including *risk-reward curves* and *R-Loss* measures for quantifying and visualizing risk-reward tradeoffs of query expansion algorithms.
- An extensive empirical evaluation, including general IR performance, risk-reward analysis, and parameter sensitivity, across 700 queries from six standard TREC collections.
- Novel budget constraints for query expansion and an algorithm for finding the k-best expansions.

More generally, an extensive development of risk-aware theoretical models, algorithms, and evaluation methods was given in the author's doctoral dissertation [8]. That work introduced the risk framework for query expansion described here and also discussed extensions to other areas of information retrieval. Recently, we note that Wang [27] applied a Markowitz-type mean-variance objective to balance risk and reward for document ranking.

4 Evaluation

In this section we give an extensive analysis showing that applying REXP to a strong, state-of-the-art baseline expansion algorithm (Indri 2.2) not only consistently and substantially reduces the worst-case performance (downside risk) of the baseline expansion algorithm across queries, but does so *without reducing its strong average performance*. In other words, REXP greatly improves the *stability* of the baseline expansion algorithm without hurting its overall effectiveness. Moreover, we show that REXP is effective when applied to *alternate* baseline expansion algorithms: most notably, REXP is effective at attenuating noise when a very poor quality baseline is used as input to REXP instead (Sec. 4.6). Furthermore, we analyze the sensitivity of the algorithms's risk-reward profile to changes in the various parameters and show that a single, consistent set of parameter settings for REXP works well for all collections in the study (Sec. 4.5).

Because our approach is a post-process that assumes little about the baseline expansion method used as input, we emphasize that the key performance question here is *not* how absolute expansion gain compares across other studies, but how much relative improvement we gain from applying REXP to an already strong baseline expansion algorithm.

We report results using standard retrieval measures, robustness histograms, and risk-reward curves. Our evaluation uses six TREC topic sets, totaling 700 unique queries: TREC 1&2 (topics 51–200), TREC 7 (topics 351–400), TREC 8 (topics 401–450), wt10g (topics 451–550), robust2004 (topics 301-450, 601–700), and gov2 (topics 701–850). We chose these corpora for their varied content and document properties. Indexing and retrieval were performed using the Indri system in the Lemur toolkit [24][17]. Our queries were derived from the title field of the TREC topics and phrases were not used. We wrapped the initial query terms with Indri's `#combine` operator, performed Krovetz stemming, and used a stoplist of 419 common English words.

For our baseline expansion method, we used the default expansion method in Indri 2.2, which first selects terms using a log-odds calculation, then assigns final term weights using the Relevance Model [16]: document models were Dirichlet-smoothed with $\mu = 1000$. We chose this baseline for its

[3]This problem has close connections to finding ambiguity groups in fault detection [30].

[4]Naturally, the actual query may contain typos, misspellings or verbal disagreement. Here we assume that alterations like spelling correction have already been performed.

consistently strong average performance: for example, in a TREC evaluation using the GOV2 corpus [11], the Indri expansion method gave a 19.8% gain in MAP over unexpanded queries, and achieved an average MAP gain of 14.4% over the six collections in this study. Indri's feedback model is linearly interpolated with the original query model weighted by a parameter α. By default we used the top 50 documents for feedback and the top 20 expansion terms, with the feedback interpolation parameter $\alpha = 0.5$ unless otherwise stated.

We set the inputs to REXP as follows. For efficiency, we limited our vocabulary V to the top $n = 100$ expansion term candidates based on their Relevance Model probability. With these Indri term scores, the entries of the vector p were set using Eq. 1. The matrices Σ, A, and g_i were also calculated dynamically for each query using the definitions given in Section 2. The entries of Σ and g_i are determined by the definition of the distance function $d(w_i, w_j)$, which in turn is defined in terms of the similarity function $\sigma(w_i, w_j)$. Here, experiments used the perturbation kernel measure [6], but as discussed earlier a simpler method such as the Jaccard similarity may also be used. The matrix A is a $|Q| \times K$ matrix, with each row being the feature vector $\phi(q_i)$ for query term q_i. There is one vector g_i for each query term $q_i \in Q$, as defined in Sec. 2.3. We set $\kappa = 1.0$ and $\gamma = 0.75$ after experimenting with a subset of queries from the TREC 1&2 and TREC 7&8 collections: in general, a unified set of parameters appears to work well across all collections in this study, and this is discussed further in Section 4.5.

4.1 Risk-reward performance

Informally, a *reward* measure reflects the quality or relevance of results for a given query, in a way that is appropriate for the task[5]. Because we are interested in ad-hoc retrieval we use Average Precision (AP) as our default reward measure. We call an *expansion failure* a case where the reward measure from applying expansion to a query is worse than the reward from the initial query results. Mathematically, we denote $R_I(Q)$ as the initial reward obtained with the query Q with no expansion, and $R_F(Q)$ as the final reward obtained when a query expansion algorithm is applied to Q.

The key aspects of a *risk* measure are: 1) that it captures *variance* or related negative aspect of retrieval performance across queries and 2) this variance/risk is based on the underlying 'reward' measure chosen. To evaluate expansion algorithms, we assume the results from the initial query represent our minimal acceptable retrieval performance: we do not want to obtain worse results than the initial query[6]. We are particularly interested in the *downside risk* of an algorithm, which we define as the reduction in reward due to expansion failures. Then the downside risk $F_{FAIL}(Q)$ for query Q is simply

$$F_{FAIL}(Q) = \begin{cases} R_I(Q) - R_F(Q) & \text{if} \quad R_I(Q) - R_F(Q) > 0 \\ 0 & \text{otherwise.} \end{cases}$$

When the reward measure is precision at the top k documents (P@k) we define *R-Loss at k* as the *net loss of relevant documents in the top-k due to failure*. When the reward measure $R_I(Q)$ is average precision we refer to this simply as *R-Loss*, setting k to the size of the retrieved document set ($k = 1000$ unless otherwise specified). Just as AP gives a combined picture of precision results averaged over multiple values of k, so the R-Loss measure gives an averaged net loss of relevant documents due to failure[7]. We use R-Loss as our default risk measure in this study.

4.2 Risk-reward curves

A risk-reward curve is generated by plotting a risk measure on the x axis, and a reward measure on the y axis, so that we can view how they trade off as the amount of expansion, controlled by the interpolation parameter α, is increased from $\alpha = 0$ (no expansion at the origin) to $\alpha = 1$ (all expansion, no initial query). The x-axis summarizes downside risk with R-Loss, the net loss in relevant documents lost due to expansion failures. To emphasize the difference over using no expansion, the y-axis summarizes the change in reward averaged over all queries, that is the *percentage MAP gain* over using no expansion, so that all curves start at the origin ($\alpha = 0$). We will typically plot in α increments of 0.1.

Risk-reward curves for both the Indri expansion baseline and the robust REXP algorithm are shown in Figure 2 for all six collections, using MAP as the reward measure with percentage MAP gain on the y-axis. The dashed line is the curve given by the strong baseline Indri expansion algorithm. The solid line is the curve of the resulting robust expansion after REXP is applied to the strong baseline Indri expansion algorithm. We enlarge the point at $\alpha = 0.5$ since this is our default setting. As discussed further in Section 4.5, all results use the same unified set of parameters that was found to work well across all collections.

We say that a tradeoff curve A *dominates* a second curve B if A is *higher and to the left* of B. An algorithm with a dominant curve gives the same or better reward for any given level of risk. It is evident that, except for one brief segment at the end of the Robust2004 curve, the REXP tradeoff curve dominates the corresponding baseline curve for every topic set. For $\alpha = 0.5$, the robust algorithm loses fewer relevant documents for all collections, while achieving comparable or higher MAP gain compared to the baseline. Also, the optimal MAP gain for REXP is always higher than the corresponding optimal baseline MAP gain.

For an alternate perspective, the risk-reward curves using Precision at 20 (P20) as reward measure instead of MAP are shown in Figure 3 for the same collections and algorithms. Note that while the baseline algorithm significantly hurts performance (negative P20 gain) for values of α close to 1 on five out of six collections, applying REXP results in a far more reliable expansion model that virtually never hurts P20 at any setting of α – only on TREC 7, at the extreme $\alpha = 1$, does it give a very small loss. Overall, applying REXP consistently (and sometimes dramatically) improves the P20 risk-reward tradeoff across all values of α for five out of six collections.

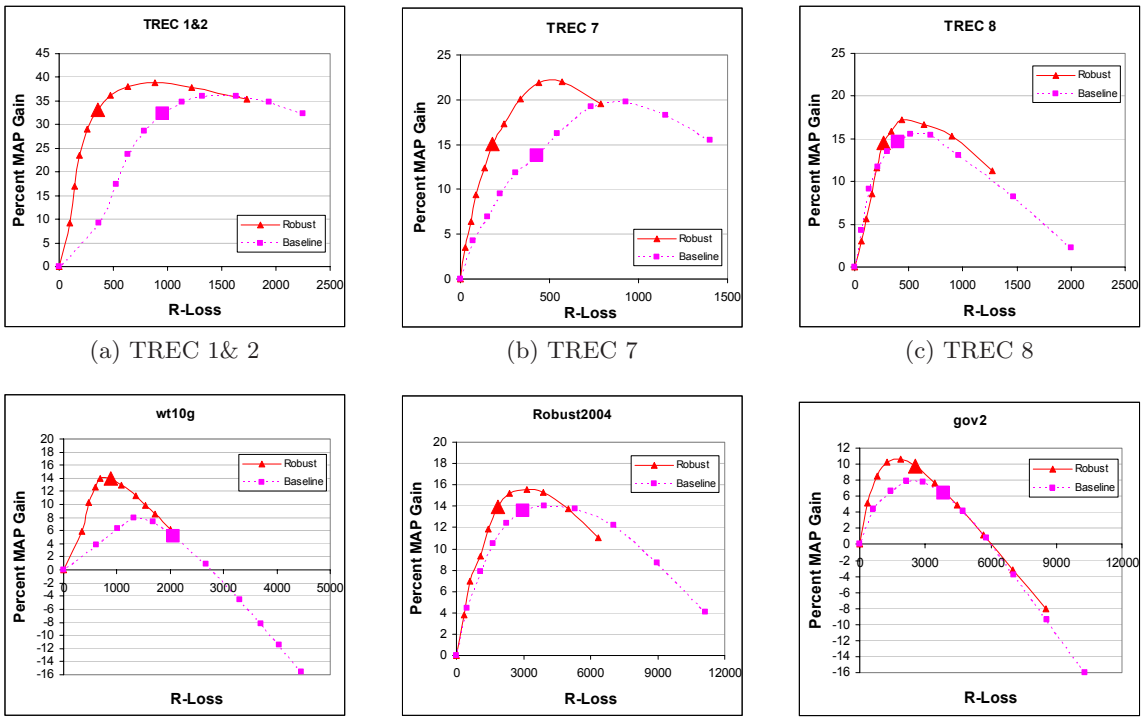
4.3 General retrieval measures

As a general summary statistic for robustness we employ a very simple measure called the *robustness index* (RI). For a set of queries Q, the RI measure is defined as: $RI(Q) = (n_+ - n_-)/|Q|$ where n_+ is the number of queries helped (i.e. with positive AP gain after expansion) n_- is the number of queries hurt, and $|Q|$ the total number of queries.

[5]For example, Web search might use the average relevance of the top-ranked document (P1) while legal applications may focus on Recall.

[6]Hypothetically, if we could reliably estimate the quality of the initial results, we could modify this assumption.

[7]This weights relevant documents equally, giving more weight to queries with more relevant documents. As with micro/macro-averaging, we could also define a normalized variant of R-Loss to weight all queries equally.



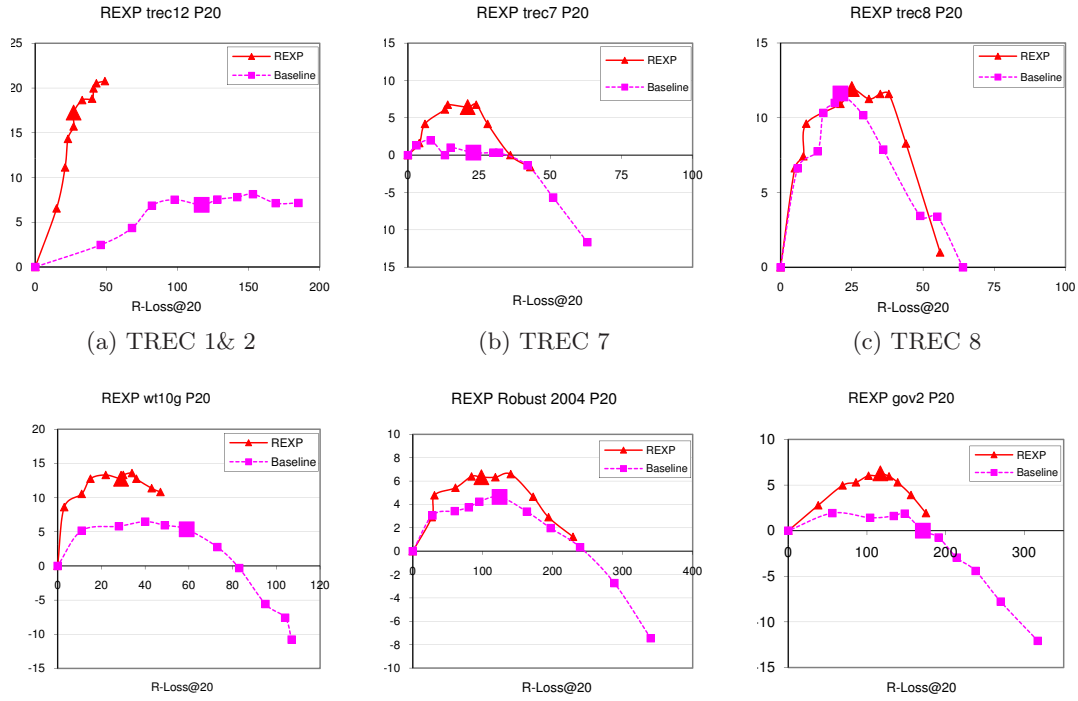
(a) TREC 1& 2 (b) TREC 7 (c) TREC 8

(d) wt10g (e) Robust 2004 (f) gov2

Figure 2: Risk-reward tradeoff curves for six TREC topic sets, showing how the robust REXP optimization step consistently improves the entire risk-reward profile for a strong baseline query expansion method (Indri 2.2). The dashed curve gives the risk-reward profile for the original strong baseline expansion method, and the solid curve shows the resulting robust REXP version. Tradeoff curves that are *higher and to the left* are better. Points are plotted in α-increments of 0.1, starting with $\alpha = 0$ at the origin and increasing to $\alpha = 1.0$. The point at the widely-used setting $\alpha = 0.5$ is show enlarged on both curves for comparison.

Collection		NoExp	Base-FB ($\alpha = 0.5$)	REXP-FB ($\alpha = 0.5$)
TREC 1&2	MAP	0.1762	0.2317 (+31.9%) $^{\mathbf{N}}$	0.2346 (+33.2%) $^{\mathbf{N,E}}$
	P20	0.4217	0.4483 (+6.94%) $^{\mathbf{N}}$	0.4945 (+17.3%) $^{\mathbf{N,E}}$
	R-Loss@20	0/366	117/366	**27/366** (-76.9%)
	RI	0	0.4844	**0.5859**
TREC 7	MAP	0.1830	0.2079 (+13.8%) $^{\mathbf{N}}$	0.2106 (+15.1%) $^{\mathbf{N}}$
	P20	0.3456	0.3467 (+0.3%)	**0.3689** (+6.8%) $^{\mathbf{N,E}}$
	R-Loss@20	0/57	23/57	**12/57** (-47.8%)
	RI	0	0.4146	**0.5610**
TREC 8	MAP	0.1920	0.2220 (+15.5%) $^{\mathbf{N}}$	0.2199 (+14.5%) $^{\mathbf{N}}$
	P20	0.3213	0.3585 (+11.8%) $^{\mathbf{N}}$	**0.3660** (+13.9%) $^{\mathbf{N}}$
	R-Loss@20	0/76	29/76	**19/76** (-34.5%)
	RI	0	0.4286	0.4286
wt10g	MAP	0.1747	0.1830 (+5.2%)	**0.1990** (+14.0%) $^{\mathbf{N,E}}$
	P20	0.2228	0.2340 (+5.4%)	**0.2512** (+12.7%) $^{\mathbf{N,E}}$
	R-Loss@20	0/158	59/158	**29/158** (-50.8%)
	RI	0	-0.0270	**0.2703**
robust2004	MAP	0.2152	0.2441 (+13.5%) $^{\mathbf{N}}$	0.2451 (+13.9%) $^{\mathbf{N,E}}$
	P20	0.3252	0.3397 (+4.5%) $^{\mathbf{N}}$	0.3458 (+6.3%) $^{\mathbf{N}}$
	R-Loss@20	0/394	124/394	**98/394** (-21.0%)
	RI	0	0.3364	0.3773
gov2 (2004–2006)	MAP	0.2736	0.2907 (+6.5%) $^{\mathbf{N}}$	**0.3004** (+9.8%) $^{\mathbf{N,E}}$
	P20	0.5214	0.5214 (+0.0%)	**0.5524** (+6.0%) $^{\mathbf{N,E}}$
	R-Loss@20	0/575	171/575	**116/575** (-32.2%)
	RI	0	0.0922	**0.2624**

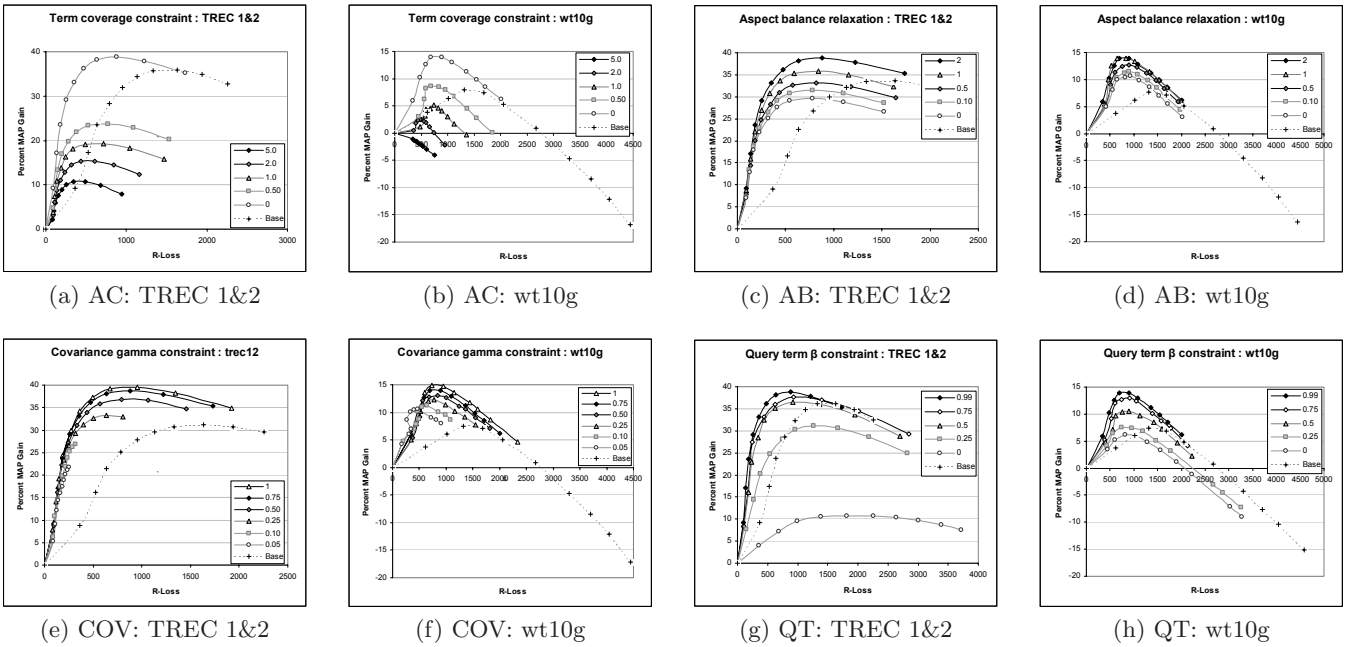
Table 1: Performance comparison of baseline (Base-FB) and robust (REXP-FB) feedback. Precision improvement shown for both the Indri expansion baseline Base-FB and the robust version of the expansion baseline REXP-FB is relative to the initial query performance. R-Loss change for REXP is relative to baseline expansion (negative change is good). For Robustness Index (RI), higher is better. Significant differences at the 0.05 level using the Wilcoxon signed-rank test are marked by **N** and **E** superscripts, for improvement over NoExp and Base-FB respectively.



(a) TREC 1& 2 (b) TREC 7 (c) TREC 8

(d) wt10g (e) Robust 2004 (f) gov2

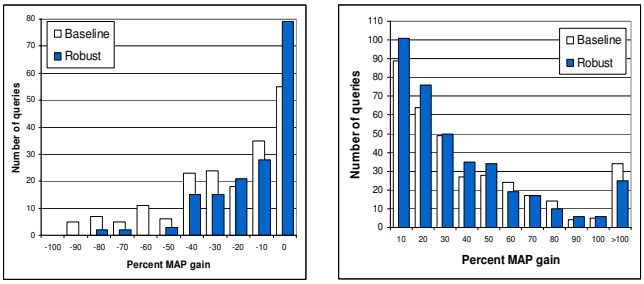
Figure 3: Risk-reward tradeoff curves for the same six TREC topic sets as Fig. 2 above, but using R-Loss@20 as the risk measure on the x-axis, and percentage gain in precision at 20 (P@20) on the y-axis as the reward measure instead of percentage MAP gain. As in Fig. 2, the point at the widely-used interpolation setting $\alpha = 0.5$ is enlarged for comparison on both curves: *higher and to the left* is better.



(a) AC: TREC 1&2 (b) AC: wt10g (c) AB: TREC 1&2 (d) AB: wt10g

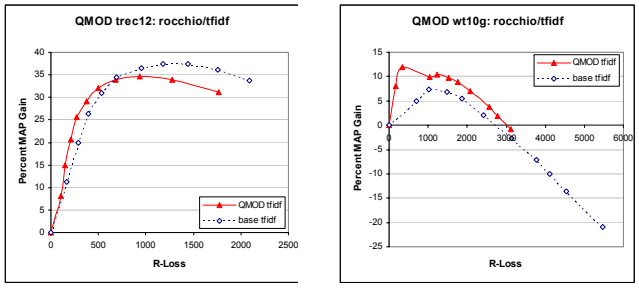
(e) COV: TREC 1&2 (f) COV: wt10g (g) QT: TREC 1&2 (h) QT: wt10g

Figure 4: The sensitivity of risk-reward tradeoff curves to varying different optimization parameters: aspect coverage (AC: ζ_i), aspect balance (AB: ζ_μ), covariance (COV: γ), query support (QT: l_i). For space reasons, two representative corpora are shown: TREC 1&2 (left) and wt10g (right). The baseline expansion tradeoff curve is also shown (dotted line).



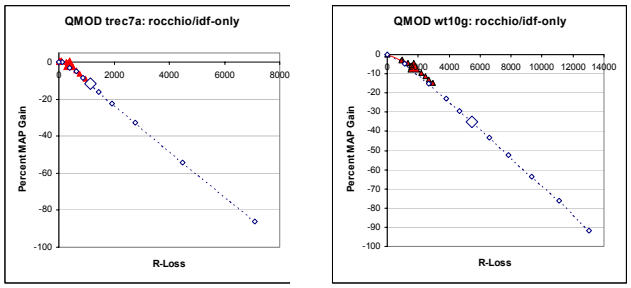
(a) Queries hurt (b) Queries helped

Figure 5: Histograms combining results for all six TREC collections, showing the robust REXP version hurts significantly fewer queries, seen by the greatly reduced tail on the left half (queries hurt). (Recall that MAP performance of REXP is also as good or better than the strong expansion baseline.) The histograms show counts of queries, binned by percent change in MAP, for the REXP algorithm (dark) and baseline (lighter).



(a) TREC 1& 2 (b) wt10g

Figure 6: The effect on risk-reward tradeoff curves of applying REXP (solid line) to a Rocchio-style expansion algorithm (dotted line) instead of the default Relevance model baseline. Tradeoff curves that are *higher and to the left* are better. Points are plotted in α-increments of 0.1, starting with $\alpha = 0$ at the origin and increasing to $\alpha = 1.0$.



(a) TREC 7 (REXP) (b) wt10g (REXP)

Figure 7: Risk-reward tradeoff curves for two representative TREC topic sets, showing the much greater tolerance of the convex REXP algorithm (solid) to noise from a poor baseline expansion algorithm (dashed). The point corresponding to $\alpha = 0.5$ for each method is enlarged for visibility. Results for other collections are similar.

Table 1 compares average precision, R-Loss, and RI statistics for the initial, baseline, and REXP feedback methods for specific choices of $\alpha = 0.5$ (the standard setting). For all six collections, at $\alpha = 0.5$ the average precision and P20 for REXP are statistically equal or superior to the Indri baseline expansion, while REXP also reduces the number of relevant documents in the top 20 lost to failures (R-Loss@20) by amounts ranging from 34.5% (TREC 8) to 76.9% (TREC 1& 2). (Note that R-Loss is relative to initial retrieval and thus always zero for the no-expansion case.) The total number of relevant documents is shown in the denominator of the R-Loss fraction. Looking at the simple fraction of net queries helped using the Robustness Index (RI), REXP at $\alpha = 0.5$ outperforms the baseline at $\alpha = 0.5$ on 5 out of 6 collections, and has equal performance for TREC 8.

4.4 Robustness histograms

Robustness histograms provide a more detailed look at how badly queries were hurt and helped by an expansion algorithm. Figure 5 gives the combined robustness histogram across the six topic sets for REXP (dark) and the baseline (light). The worst failures – cases where a query's average precision was hurt by more than 60% – have been virtually eliminated by the REXP algorithm, while the upside gain distribution remains very similar to the baseline gains. The most noticeable differences in gains are a reduction in the highest category (more than 100% AP gain) and an increase in the lowest gains (0 to 10%). Both of these are due to the selective expansion mechanism of the REXP algorithm: queries that are deemed too risky to expand are not expanded, resulting in a zero AP gain.

4.5 Parameter and constraint sensitivity

Because the REXP program uses several parameters, we provide a detailed study of how different choices in this parameter space affect performance and how *sensitive* the quality of the expansion solution is to changes in the parameters. We also show that there is a single, consistent choice of parameters that works well for all six collections we tried.

Figure 4 summarizes the sensitivity of tradeoff curves to different constraint parameter values. The most dominant tradeoff curves were obtained using an intermediate mix of risk and reward objectives, with all constraints active. Some constraints had a more dramatic effect on the tradeoff curve than others. Query support (l_i) was a highly influential constraint: it had a strong effect on MAP gain, but little effect on risk. Conversely, the use of off-diagonal Σ_{ij} covariance entries (γ) had a larger effect on risk reduction (and a weaker effect on MAP). Activating both of these together resulted in most of the improvement in the REXP tradeoff curve. Other constraints such as the aspect balance constraint ζ_μ were less critical but acted to further shrink the risk of the tradeoff curve with little reduction in MAP. Increasing the aspect coverage parameter ζ_k also acted to increase the conservativism of the solution. The role of κ is similar to that of the interpolation α, controlling the mix between a solution close to the original query and one using all expansion terms. We used a setting for REXP parameter values that is effective on all collections: high query support ($_i = 0.95$), moderately relaxed aspect balance ($\zeta_\mu = 2.0$), minimal aspect coverage constraint ($\zeta_k = 0.1$ for all q_k), medium covariance regularization ($\gamma = 0.75$) and equal objective weighting ($\kappa = 1.0$).

4.6 Alternative expansion baselines

To show the generality of REXP's black-box approach and its tolerance to noise, we replaced the Indri baseline algorithm with a strong Rocchio-type method [22] based on a vector-space model, and a noisy idf-only version.

Rocchio-type. We used a Rocchio-style vector space baseline in which the top k document vectors were given equal weight and used a *tf.idf* representation. For space reasons the tradeoff curves for two representative collections are shown in Figure 6: others are similar. As it did with the Relevance model baseline, REXP dominates the Rocchio *tf.idf* baseline for wt10g. It also reduces R-Loss for trec12, while keeping average MAP gain comparable.

High-noise Rocchio. When faced with a poor-quality expansion baseline, a good selective algorithm should usually avoid expansion altogether and revert to the original query: REXP does indeed behave exactly this way. This baseline method is a noisy version of the Rocchio scheme that ignores term frequency (tf) and uses only *idf* in the term weighting, which results in a noisy expansion model dominated by rare terms that are poor discriminators for relevance. The results for the same two representative collections, TREC 7 and wt10g, are shown in Figure 7. This *idf*-only baseline has terrible performance, with MAP loss at $\alpha = 1.0$ worse than -80%. However, REXP using this baseline almost completely attenuates the damage by scaling back to very conservative expansion. At $\alpha = 0.5$, for TREC 7a, MAP loss is reduced from -11.8% to almost zero (0.88%) with reduction in R-Loss from 1136 to 390. For wt10g, MAP loss is reduced from -35.1% to -6.1% with reduction in R-Loss from 5485 to 1703. The other four standard collections have similar results: REXP MAP loss at $\alpha = 0.5$ is between 0% and -5%, versus baseline MAP loss of -20% to -40%.

5 Discussion

Based on our evaluation and observations, we believe there are three distinct capabilities that any expansion algorithm should have to be both reliable and effective. First, uncertainty in the data should be captured and applied to adjust the conservativeness of the solution from query-to-query. In our model this done via the robust problem's uncertainty set $\mathcal{U}$. Second, a ŚhardŠ selection process should eliminate implausible models completely: if necessary, all hypotheses except the observed query may be rejected. We implement this property by defining a feasible set using hard constraints, such as query support. Such constraints are very effective in attenuating noise when the baseline expansion model is very poor. With sparsity or budget constraints included, this dynamically chooses the number of top-k final expansion terms (including zero terms), rather than forcing us to choose a fixed k in advance. Third, a final process searches for the optimal model based on the objective function over the remaining good (feasible) models, effectively performing a kind of model combination over the space of feasible solutions. Current algorithms implement some of these, but beyond the present work, no existing algorithms that we are aware of effectively address all three requirements at once. The result of combining them is a reliable, selective, effective expansion algorithm.

6 Conclusions

This paper contributes fundamental new tools for the development and evaluation of query expansion algorithms. By applying concepts from computational finance to cast query expansion as a robust constrained convex optimization problem, we can bring the full power of this technology to bear on expansion problems in order to tradeoff risk and reward and handle domain constraints that would be difficult or impossible using traditional expansion methods. We showed how we can model our uncertainty in important constraints by defining uncertainty sets and minimizing the optimal loss over the uncertainty set. This leads to conservative solutions by using robust optimization versions of the basic program, which turn out to have a simple, efficient analytical form. While most proposed improvements to query expansion only apply to a particular retrieval model, our algorithms treat the retrieval model as a black box which could be implemented using vector space models, inference networks, statistical language modeling, or other approaches. Thus, the approach we have described is broadly applicable.

We also introduced risk-reward tradeoff curves, which we believe should be a standard evaluation method for query expansion algorithms. With these curves and other evaluation measures, we showed how the downside risk of existing algorithms can be significantly improved, with no loss of average upside gain. Furthermore, we showed that our robust optimization method almost completely attenuates the damage caused by a poor baseline algorithm.

We also described extensions such as budget constraints and k-best expansions that fit easily into this framework. This work opens new research directions to explore further constraints and objectives, such as biasing expansions with personalization models or implicit and explicit relevance feedback. Finally, further gains may be possible with data-driven learning of constraints or objective parameters.

Acknowledgements

We thank Stephen Boyd for valuable discussions on related work and real-time implementation issues; Jamie Callan, William Cohen, Susan Dumais, and John Lafferty for their extensive feedback on many aspects of this research; and Paul Bennett and Jaime Teevan for their helpful comments and editing suggestions.

7 References

[1] G. Amati, C. Carpineto, and G. Romano. Query difficulty, robustness, and selective application of query expansion. In *ECIR 2004*, pages 127–137.

[2] A. Ben-Tal and A. Nemirovski. Robust solutions of uncertain linear programs. *OR Letters*, 25:1–13, 1999.

[3] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, USA, 2004.

[4] C. Buckley. Why current IR engines fail. In *Proceedings of SIGIR 2004*, pages 584–585.

[5] G. Cao, J.-Y. Nie, J. Gao, and S. Robertson. Selecting good expansion terms for pseudo-relevance feedback. In *Proceedings of SIGIR 2008*, pages 243–250.

[6] K. Collins-Thompson. Robust word similarity estimation using perturbation kernels. In *Proceedings of the International Conference on Theoretical Information Retrieval (ICTIR) 2009*.

[7] K. Collins-Thompson. Estimating robust query models using convex optimization. In *Advances in Neural Information Processing Systems 21 (NIPS)*, pages 329–336, 2008.

[8] K. Collins-Thompson. *Robust model estimation methods for information retrieval*. PhD thesis, Carnegie Mellon University, 2008.

[9] K. Collins-Thompson and J. Callan. Query expansion using random walk models. In *CIKM 2005*, pages 704–711.

[10] K. Collins-Thompson and J. Callan. Estimation and use of uncertainty in pseudo-relevance feedback. In *Proceedings of SIGIR 2007*, pages 303–310.

[11] K. Collins-Thompson, P. Ogilvie, and J. Callan. Initial results with structured queries and language models on half a terabyte of text. In *TREC 2005*. NIST Special Publication.

[12] E. N. Efthimiadis. *Annual Review of Information Science and Technology*, Vol. 31, Chapter: Query expansion, pages 121–187. 1996.

[13] S. P. B. Emmanuel J. Candès, Michael B. Wakin. Enhancing sparsity by reweighted ℓ_1 minimization. *Journal of Fourier Analysis and Applications*, 2008.

[14] D. Harman and C. Buckley. The NRRC Reliable Information Access (RIA) workshop. In *SIGIR 2004*, pages 528–529.

[15] G. Lanckriet, L. El Ghaoui, C. Bhattacharyya, and M. Jordan. A robust minimax approach to classification. *Journal of Machine Learning Research*, 3:555–582, 2002.

[16] V. Lavrenko. *A Generative Theory of Relevance*. PhD thesis, University of Massachusetts, Amherst, 2004.

[17] Lemur. Lemur toolkit for language modeling & retrieval. 2002.

[18] H. M. Markowitz. Portfolio selection. *Journal of Finance*, 7(1):77–91, 1952.

[19] Q. Mei, D. Zhang, and C. Zhai. A general optimization framework for smoothing language models on graph structures. In *Proceedings of SIGIR 2008*, pages 611–618.

[20] D. Metzler and W. B. Croft. Latent concept expansion using markov random fields. In *SIGIR 2007*, pages 311–318.

[21] M. Mitra, A. Singhal, and C. Buckley. Improving automatic query expansion. In *SIGIR 1998*, pages 206–214, 1998.

[22] J. Rocchio. *The SMART Retrieval System*, chapter Relevance Feedback in Information Retrieval, pages 313–323. Prentice-Hall, 1971. G. Salton, ed.

[23] A. Smeaton and C. J. van Rijsbergen. The retrieval effects of query expansion on a feedback document retrieval system. *The Computer Journal*, 26(3):239–246, 1983.

[24] T. Strohman, D. Metzler, H. Turtle, and W. B. Croft. Indri: A language model-based search engine for complex queries. In *Proc. Int. Conf. on Intel. Analysis*, 2004.

[25] T. Tao and C. Zhai. Regularized estimation of mixture models for robust pseudo-relevance feedback. In *SIGIR 2006*, pages 162–169.

[26] J. Teevan, S. T. Dumais, and E. Horvitz. Personalizing search via automated analysis of interests and activities. In *SIGIR 2005*, pages 449–456.

[27] J. Wang. Mean-variance analysis: A new document ranking theory in information retrieval. In *ECIR 2009*, pages 4–16, 2009.

[28] M. Winaver, O. Kurland, and C. Domshlak. Towards robust query expansion: model selection in the language modeling framework. In *SIGIR 2007*, pages 729–730.

[29] J. Xu and W. B. Croft. Improving the effectiveness of information retrieval with local context analysis. *ACM Trans. on Information Systems*, 18(1):79–112, 2000.

[30] A. Zymnis, S. Boyd, and D. Gorinevsky. Relaxed maximum a posteriori fault identification. *Signal Processing 89*, 989–999, 2009.