Statistical Query Transformations for Question Answering in the Web

Ilya Boyandin, Igor Nekrestyanov

October 2003

Аннотация

This paper examines the problem of transforming questions posed by users of a domain-independent factual search system in the Web from natural language into queries for a general-purpose search engine.

We analyze the performance of a statistical query transformation algorithm based on the QASM model [14]. The search quality evaluation is performed on real Russian-language factual queries from the Yandex search engine log.

1 Introduction

Factual search is a variant of text search with reduced granularity [19]. Unlike classical search problems, factual search requires finding not documents on the query topic, but precise and concise answers to specific questions formulated in natural language. For example, for the question: "Who was the first cosmonaut?" an ideal factual search system should return a single answer: "Yuri Gagarin".

Unlike question-answering systems with active knowledge acquisition, domain-independent factual search systems are not required to perform logical inference - the system only needs to extract from the dataset a short text fragment that answers the question. Therefore, the result of such a system strongly depends on the text dataset being searched. For example, if searching for an answer to the same question about the first cosmonaut in a collection of texts about American astronauts, then "Alan Shepard" could well be the correct answer.

Currently there are no industrial systems on the Internet capable of automatically processing factual queries with acceptable quality. But the need for processing such queries undoubtedly exists. According to analysis of the Excite search engine log[1] about 8% of user queries are correct English language questions, of which about 44% are factual [1]. Users of the Russian-language search engine Yandex also frequently formulate their queries as correct questions [20].

The relevance of factual search drives active research in this area. In particular, within the TREC conference there has been a separate track for several years dedicated to experimental evaluation of factual search systems [16].

[1] for December 20, 1999

The huge volume of constantly updated and growing textual information on the Internet gives factual search systems the potential ability to find answers to much more diverse factual questions than is possible within a closed document collection.

However, Internet data, due to the lack of centralized control over publication, has several characteristics that complicate the search task: unstructured, heterogeneous, contradictory. On the other hand, there is also redundancy and repetition of data on the Internet: the answer to the same question, formulated differently, may be contained in dozens of documents. As noted by several researchers, this redundancy can be used to improve the quality of factual search [8, 6], which is indirectly confirmed by the results of experiments to identify the dependence of factual search quality on collection size [4, 5].

Query processing in a factual search system is usually done in several stages [12]:

1. **Preliminary**
   At this stage, question classification is usually performed, based on the results of which, the query is formulated and possible answer forms are determined.

2. **Question transformation**
   The question is transformed into one or more search engine queries, so that the found documents as accurately as possible represent documents in the collection that contain possible answers.

3. **Text search**
   The search engine, using traditional information retrieval methods, finds collection documents matching the formulated queries.

4. **Answer extraction**
   From the documents found by the search engine, small text fragments containing the most likely answers are selected. Systems try various ways to verify the correctness of each possible candidate answer and discard "unconvincing"ones.

Existing general-purpose search engines on the Internet can be effectively used by a factual search system at the text search stage to find documents with possible answers [13]. For example, the query "`Nabokov /1 !was born`" sent to Yandex, gives five correct answers (in 1899) out of the first five, while among the first five documents obtained for the query "`When was Nabokov born?`", only one contains the correct answer[2]. The quality of transforming a question into a search engine query largely determines the overall search quality. Indeed, if no document containing the sought answer is found using the constructed queries, subsequent stages cannot fix the situation; if several found documents contain the correct answer, this increases the probability of its selection by the system at the answer extraction stage[3].

[2]The described experiment with Yandex was conducted on March 15, 2003

[3]In many systems (for example, in the Mulder system [12]) scores for possible answers at

Question	Question Type
What is the name of logotherapy's creator?	PERSON
In what year was the Mausoleum built in Berlin?	DATE
Where is the Taj Mahal located?	LOCATION
What is the distance from Abu Dhabi to Agra?	DISTANCE

Таблица 1: Examples of question types

The goal of this work was to study the possibility of effective question transformation based on statistical methods. To date, the best factual search results are provided by systems that actively use natural language processing methods, but as shown by the example of the Tritus system[1], systems using statistical methods and very simple language knowledge can achieve significant results.

The use of statistical approaches often allows significantly reducing computational complexity of query processing and, consequently, improving system scalability. Another important advantage of the statistical approach is avoiding the need to use high-quality linguistic resources, availability of which, for example for Russian, is limited.

As a starting point for our research, we used the QASM algorithm [14], which learns question transformations on a set of questions and answers. Starting with the QASM model, we tried to find answers to several questions: what is the maximum achievable transformation efficiency in this model? how different is the effectiveness of operators and properties? what factors influence the optimal choice?

The further exposition has the following structure: Section 2 presents a review of related work, Section 3 describes the basic QASM algorithm, and Section 4 describes our modifications to the algorithm. Section 5 outlines the most important characteristics of the factual search system prototype, analysis of experimental results with which is presented in Section 6.

2 Related Work

Query transformation is actively used in many information retrieval tasks. By transformation goal type they can be divided into two main classes:

- **Query "translation**

  This group includes transformations intended to express the same query in another form while maximally preserving properties of the original query. For example, this class includes query transformations by mediators in metasearch systems (where each search server may have its own query language with specific syntax and semantics) [11] or in multilingual search

the answer extraction stage are calculated such that the probability of extracting an answer that appears more frequently in found documents increases.

systems (where the original query can be automatically reformulated in another language) [17].

- **Query "refinement**

  Transformations of this type are initially oriented towards changing query properties, that is, its semantics. The goal of this change is usually to obtain a new query version that better describes the information need behind the original query. Such transformations, for example, are often used together with relevance feedback mechanisms for iterative refinement of user needs by the search system [2].

In the context of domain-independent factual search, transformations of the second type are of particular interest, since general-purpose search engines are not designed for searching answers to natural language questions, and in order to correctly formulate the user's real information need in the search engine's query language, it is necessary to change the semantics of the original question. For example, some words contained in a question do not necessarily have to be contained in the correct answer to that question: they may simply be absent, present in a different form or be replaced by synonyms or generalizing concepts.

Question transformations in factual search systems can usually be represented as a sequence of operations, such as: adding, removing or replacing words, phrases or adding search engine syntax operators, morphological word transformations, etc. (see, for example, [12]).

The choice of transformation operations applied to a question is based on various characteristics of the question or individual words involved in the question. The most important property of a question is its type. Sets of question types used by different systems, and methods for determining them differ, but in most systems the question type is defined by the question object (see Table 1), which is determined by question words (see, for example, [1]) or using more complex syntactic and semantic parsing of the question (see, for example, [10] or [3]). The characteristics of individual question words that are used to determine transformations may include: word role (e.g., interrogative or not), part of speech, word significance (evaluated based on its usage frequency) and others. [14]

Transformation rules can be predefined in the system in advance with varying degrees of generalization. For example, in [15] simple queries are formulated, consisting of the most "important"question keywords by statistical properties. This approach provides low precision but fairly good recall[4].

In the Falcon system [10] the optimal degree of query relaxation is determined dynamically. After receiving search results for the initial query, the system formulates a relaxed query version (removing some words) if too few results are found, or formulates a stricter query (adding terms) if there are too many results.

[4]Note that this approach significantly increases the load on subsequent steps of factual query processing. Although in this case the system receives more documents containing the correct answer, it also receives a significantly larger number of inappropriate documents, which can mislead it.

In the AskMSR system [8] transformation rules are set manually. Sufficiently strict rules created manually can provide high search precision. However, since strict rules are usually narrowly specialized, that is, applicable in a very limited set of cases, creating an exhaustive set of such rules that takes into account all features of natural language is hardly possible.

In recent years much attention has been paid to research on approaches that automatically learn query transformations (see, for example, [9]), including for factual search.

The Tritus system [1] learns question transformation rules on a set of "frequently asked questions"and answers to them. During training, the system first tries to find important phrases most frequently occurring in text fragments containing answers to questions of each type, weighting phrases using weights similar to $tfidf$, and builds transformation rules using these phrases. Then the system evaluates the quality of all obtained transformations by applying them to all questions of the corresponding type from the training set, sending queries to the search engine and evaluating the proximity of the first n found documents to the answer known to it, and selects and remembers only the best transformations.

3 The QASM Algorithm

The probabilistic QASM (Question Answering using Statistical Models) [14] algorithm learns transformations that are compositions of atomic transformations.

The algorithm's task is to construct from the input question a sequence of atomic transformations whose composition when applied to it gives the best query[5]. The query construction procedure is iterative: at each step an atomic transformation is chosen that improves the query; iterations continue while improvement is possible.

In order for the QASM algorithm to be able to construct question transformations, it needs to go through a training phase, during which it identifies patterns connecting query characteristics and successful transformations.

In general, the task of choosing an atomic operator can be considered as a classification task: the algorithm must decide which class to assign the query to at a given step, and apply the atomic operator corresponding to the chosen class.

3.1 EM Algorithm

The QASM learning algorithm is based on the well-known statistical *Expectation Maximization* (EM) algorithm — an iterative algorithm for finding maximum likelihood estimates. This algorithm is often used in problems with incomplete data. In our case during training only the set of questions and answers is known,

[5]In this article the term *question* is used to denote the original user question in natural language, and the term *query* is most often used to denote the transformed version of the question that is sent to the search engine.

while the transformations themselves that give the best result for each question are unknown.

The EM algorithm consists of the following steps:

1. estimate unknown parameters (using algorithm-available measurement results and model data),
2. modify the algorithm model (based on estimates obtained in step 1),
3. if local maximum not reached, go to step 1.

It is known that the EM algorithm ensures improvement of obtained estimates with each step and eventually converges [7].

3.2 Training

QASM learns on a set of questions and answers, trying to find for each question in the test set the best sequence of atomic transformations. Formally the task can be stated as follows.

Let $\mathcal{A}_1, \ldots, \mathcal{A}_l$ be a fixed set of functions (*query properties*) that map a query to an integer. The ordered set of numerical values of all these functions for a specific query q is called the context $\mathcal{C}(q)$ of the query:

$$\mathcal{C}(q) := (\mathcal{A}_1(q), \ldots, \mathcal{A}_l(q))$$

Let $\mathcal{O}_1, \ldots, \mathcal{O}_m$ be a fixed set of *atomic query transformation operators* that map query q to new query $\mathcal{O}_i(q)$, and $F(q)$ be some function evaluating *actual quality* of the query, which can use information about documents found by this query (see section 5.3).

Note that only the $Identity$ operator that maps a query to itself ($Identity(q) = q$) is fixed. The choice of other operators, properties and function $F(q)$ does not affect the general algorithm and depends on its implementation.

The learning algorithm must determine mapping T that associates any set $\mathcal{C}(q)$ of query q property values with number r of the atomic operator leading to the highest quality transformation.

In other words, the learning algorithm builds classifier T that maps each valid context to a class corresponding to one of the atomic operators, the best one for that context.

The mapping T in the QASM algorithm is defined by matrix

$$\Theta = \{p(\mathcal{O}_i|\mathcal{C}_j)\}_{i,j}$$

of probabilities $p(\mathcal{O}_i|\mathcal{C}_j)$ of applying operator $\mathcal{O}_i$ to a query defining context $\mathcal{C}_j$:

$$T(\mathcal{C}_j) = \underset{i}{argmax}(\Theta_{i,j})$$

where $\mathcal{C}_j$ is a valid context with number j (the number of all valid contexts is finite, so they can be numbered). Moreover,

$$\forall j : \sum_{i=0}^{m} p(\mathcal{O}_i|\mathcal{C}_j) = 1$$

Matrix Θ is initialized with uniform distribution across all operators. Then, the learning algorithm is sequentially executed for each question in the training set on the same matrix Θ:

1. Apply each operator to the question, evaluating quality of resulting queries (correct answer to original question is known). If highest quality is provided by *Identity* operator, then processing of current query is complete. Otherwise — based on probability distribution given in Θ for query context, choose operator that will be applied to query.

2. Apply operator chosen in first step to query. Modify Θ using information about query quality obtained in first step:

   - operators are ordered by decreasing quality of resulting queries (for given query q),
   - probabilities $p(\mathcal{O}_i|\mathcal{C}(q))$ in row Θ corresponding to $\mathcal{C}(q)$ are multiplied by $oprank_i^{-1}$, where $oprank_i$ is rank of i-th operator (assigned to it as result of ordering),
   - matrix rows are normalized so sum of values in row equals 1.

3. If change in matrix Θ on this iteration does not exceed given threshold ($\delta(\Theta) < \varepsilon$), then processing of current query is complete, and constructed matrix Θ is result of algorithm training. Otherwise cycle repeats from first step.

3.3 Question Transformation

After training is complete, system can transform questions that were not in training set. The QASM question transformation algorithm [14] maps a question to transformed query, sequentially computing query context and applying most probable (according to distribution given in Θ) operator to query, recalculating context and choosing operator again, and so on until at some point this operator becomes *Identity* (or another operator that does not change given query). Thus, system will output query q_s, where

$$q_k = \mathcal{O}_{T(\mathcal{C}(q_{k-1}))}(q_{k-1}),$$

$k \in 1 : s$, and q_0 is original question.

4 QASM Modifications

In addition to original QASM algorithm described in previous section we considered two alternative approaches working within same model.

4.1 Optimal QASM (oQASM)

This QASM modification is intended to evaluate maximum achievable result when using QASM model. It is assumed that this algorithm always returns best question transformation that can be constructed with given set of atomic operators.

Practical implementation of this algorithm was based on complete enumeration of all possible transformations for each test set question and selection of one giving best result.

Note that this evaluation is not upper bound in general case, since it is quite likely that better result can be achieved when using some other set of atomic operators.

4.2 Multiple QASM (mQASM)

Original QASM generates only one transformed query from input question. However our experiments showed that due to irregularity of Internet data, even for very similar queries (indistinguishable from trained model's perspective) best result can be provided by different transformations. For example, such queries are questions: "`Who was Nobel Peace Prize laureate in 1975?`" and "`Who was Nobel Peace Prize laureate in 1979?`".

One of most important reasons for this is poorly predictable selectivity of specific query. Transformation chosen by algorithm can give excellent results on one question and, at same time, lead to query with zero selectivity for another query with very similar properties. Reverse effect is also possible — too imprecise (due to high selectivity) query.

For this reason it is natural to consider such extension of QASM that instead of one most useful query selects most useful subset of queries from set of all possible transformations of original question. However, solving this optimization problem in general case seems very complex due to impossibility of well predicting "intersection"of different queries. Therefore we used heuristic assumption that most useful subset of queries is subset of all transformations with predicted usefulness exceeding some threshold value. The mQASM algorithm is based on this heuristic.

Queries constructed by mQASM algorithm are ordered by decreasing predicted selectivity (see section 5.4) and sequentially executed. First, most strict of not yet sent queries is sent to search engine. Based on number of documents found by query, search quality is evaluated[6]. If acceptable level of search quality is achieved — that is, sufficient number of documents found — before all queries are sent, then remaining queries can already not be executed.

More formally, mQASM generates from question q, using matrix Θ built during training phase, all possible queries $\hat{q}$, obtained by sequential application of atomic operators to question, for which *selection probability* — $P(\hat{q})$ — exceeds

[6] Similar idea is used in Falcon system [10], where depending on number of documents found by query, query can be relaxed (if too few) or strengthened (if too many) and re-executed.

some threshold γ:

$$P(\hat{q}) := \max \prod_{k=0}^{r} p(\mathcal{O}_{s_k}|\mathcal{C}(q_{k-1})) \geq \gamma \quad (1)$$

where $q_0 = q$, $q_k = \mathcal{O}_{s_k}(q_{k-1})$ and $\hat{q} = q_{s_r}$. Maximum is taken over all sequences of atomic operators $(\mathcal{O}_{s_1}, \ldots, \mathcal{O}_{s_r})$, whose composition when applied to original question q gives query $\hat{q}$ (there may be more than one such sequence). $P(q)$ is predicted usefulness of query.

Generated queries form set $\hat{Q} = \{\hat{q}|P(\hat{q}) \geq \gamma\}$. For each $\hat{q} \in \hat{Q}$ significance estimate $w_{\hat{q}}$ is calculated, which determines order of query execution. Queries are executed in order of decreasing significance, until they run out or sufficient number of documents is collected (denote it N_{suff}).

Results a_i of query execution are combined into single set (maximum number of considered answers to each query is bounded above by same constant N_{suff}) and ordered by weight w_{a_i}, calculated as:

$$w_{a_i} = \frac{N_{\text{suff}} - rank_{a_i} + 1}{N_{\text{suff}}} * w_{\hat{q}}$$

where a_i is one of results of answer to query $\hat{q}$; $rank_{a_i}$ is ordinal number of a_i in list of answers to query $\hat{q}$ (that is, rank assigned to document a_i by search engine for query $\hat{q}$). If same result was obtained by several queries, then highest of its weights is taken as its weight in final combined set.

Thus, document weight is higher the closer document is to beginning of final list and higher significance estimate of query by which it was obtained.

5 System Prototype

To conduct experimental evaluation of algorithms we implemented prototype of factual search system for Web. Yandex[7] was used as base search engine. Part of described experiments we also conducted using Google[8].

5.1 Atomic Operators

We considered following atomic query transformation operators:

- *Identity* operator that maps query to itself.
- Several word removal operators: stop word removal, question word removal and operators removing words with frequency exceeding certain level.
- Gluing operators: between adjacent query words Yandex query syntax operator "`/n`" is inserted, prohibiting Yandex from returning documents where query words are more than `n` words apart from each other.

[7] http://www.yandex.ru
[8] http://www.google.com

- Morphological analysis cancellation operator: exclamation mark is placed before each query word, prohibiting Yandex from returning documents where given word is present only in other morphological forms.

Probably operators replacing words with synonyms or generalizing concepts (and also adding words), similar to those used in [14], could improve search quality, but implementation of these operators requires use of high-quality Russian-language synonym dictionaries or Russian-language thesaurus.

Note that we considered Yandex query language specific operators to expand operator set, and as it turned out, application of these operators positively affects work quality. However algorithms themselves are not tied to any specific query language or search engine.

5.2 Query Properties

In [13] it is shown that certain properties of natural language questions, namely: question type (PERSON, LOCATION etc.), number of words in it and number of proper names, affect ability of search engines to answer them, and questions with same values of these properties can be processed similarly.

In our prototype following query properties were used: question type, number of words in query, number of proper names, indicators of applying gluing and morphological analysis cancellation operators.

5.3 Search Quality Assessment

Following metrics are often used to evaluate quality of answer to factual query:

- **MRR** (Mean Reciprocal Rank) [16]:
  MRR value for one query q equals:

  $$MRR(q) = r^{-1},$$

  where r is rank of first document containing correct answer to question, returned by system among first five ($r = 0$ if among first five there is none containing correct answer).

  I.e. $MRR(q)$ can take one of six values: (1, 0.5, 0.33(3), 0.25, 0.2, 0), depending on which position among first five document with correct answer is returned.

- **TRDR** (Total Reciprocal Document Rank) [14]:
  This metric is calculated as:

  $$TRDR(q) = \sum_{i=1}^{n_{\text{corr}}} r_i^{-1}$$

  where n_{corr} is number of documents containing correct answer among first N_{eval}, returned by search engine for query q; r_i is rank of i-th document containing correct answer.

To evaluate overall search quality by MRR and TRDR metrics their average values are calculated over all test set questions.

QASM and mQASM algorithms during training use function evaluating actual query quality $F(q)$ (see section 3.2). When choosing metric used as $F(q)$ we were guided by following considerations.

When using MRR it is assumed that it is very important for user that correct answer is first in document list, therefore MRR is higher when system returns only one correct answer but in first place, than when all answers except first in results list are correct. But in case when document search results by queries formulated by system at transformation stage are passed to another system component for answer extraction, number of documents containing correct answer can be important, since many systems when extracting answers use redundancy and tend to choose answers that appear more frequently in search results. Therefore TRDR metric, differing from MRR in that when calculating it not only first correct answer is considered but also subsequent ones, is better suited for evaluating quality of query transformation, while MRR is better suited for evaluating search quality at output of full-fledged factual search system that extracts answers from found documents. In particular, our experiments showed 5% deterioration in search quality by MRR metric and 8.4% by TRDR metric when using MRR during training compared to TRDR.

Therefore to evaluate actual query quality we used TRDR metric:

$$F(q) = TRDR(q).$$

5.4 Query Selectivity Assessment

Assessment of relative selectivity of query constructed by mQASM algorithm in our prototype was performed as follows.

Each atomic operator was associated with some selectivity coefficient s_j, greater or less than 1. *Identity* operator applied to query does not change its selectivity (therefore selectivity coefficient of *Identity* operator equals 1), removal operators increase, others decrease.

Weight of query $\hat{q}$ was determined by formula:

$$w_{\hat{q}} = \prod_j s_j^{-1}$$

where s_j are selectivity coefficients of atomic operators, by sequential application of which to original question query $\hat{q}$ was obtained.

As it turned out, assessment of real values of operator selectivity coefficients is too laborious due to some features of Yandex[9], therefore prototype used heuristic values of these coefficients:

- 1 for *Identity* operator,

[9]Namely, because Yandex distinguishes several levels of matching found documents to query: strict and non-strict.

Approach	MRR	TRDR	Answers
Yandex	0.436	0.938	31 (77.5%)
QASM	0.498 (+14.2%)	0.992 (+5.8%)	29 (72.5%)
mQASM	0.519 (+19.0%)	1.155 (+23.1%)	33 (82.5%)
oQASM	0.678 (+55.3%)	1.457 (+55.2%)	35 (87.5%)

Таблица 2: Overall search quality

- from 1.05 to 2 for different word removal operators,
- 0.7 and 0.8 for two gluing operators and 0.8 for morphological analysis cancellation operator.

5.5 Query Significance Assessment

mQASM algorithm (see section 4.2) uses significance estimates $w_{\hat{q}}$ of query $\hat{q}$ to determine order of query execution, in addition significance estimates of queries affect final weights of search results.

We considered several different query weighting options:

- **Equal weights**
  All queries are assigned same weight (equal to 1).
- **Selection probability estimate**
  In this case weight $w_{\hat{q}}$ of query $\hat{q}$ is considered equal to value $P(\hat{q})$ determined by formula 1.
- **Selectivity estimate**
  In this case weight of query $\hat{q}$ was assigned selectivity estimate of $\hat{q}$.

In each of these options weights were normalized so that highest query weight equals 1.

6 Experimental Analysis

The goal of experimental analysis was to study the behavior of QASM and mQASM algorithms compared to the maximum achievable result, to determine the factors that influence the optimal choice.

6.1 Dataset

For system training, a set of 60 (question, answer) pairs of type PERSON was used, of which 30 were obtained from Yandex query logs, and 30 were artificially created.

Overall effectiveness was evaluated on a set of 40 questions of the same type (all 40 were obtained from Yandex query logs). The question sets for training and evaluation did not overlap.

The decision to limit to one type was due to the complexity of creating high-quality training and test question sets. There are no other fundamental limitations preventing evaluation of other types of factual questions within the described prototype.

Note that for queries of other types, the observed patterns are likely to change.

6.2 Search Quality Evaluation Criteria

Three metrics were used to evaluate search quality:

- MRR metric,
- TRDR metric,
- number of questions for which the system found the correct answer.

Evaluation was performed automatically: a document returned by the system was counted as a correct answer if it matched one of several regular expressions predefined for each question in the test and training sets.

When calculating all metrics, only the first 20 documents returned by the system were checked for the presence of correct answers (i.e. $N_{\text{eval}} = 20$).

6.3 Overall Search Quality

Table 2 shows the results of overall search quality evaluation obtained using the QASM, mQASM and oQASM algorithms on the original dataset (i.e. with one of the partitions of the question set into training and evaluation questions).

For comparison, it also shows search quality estimates obtained when executing unmodified questions from the test set as Yandex queries.

As can be seen from the table, the maximum achievable result (oQASM) exceeds the result obtained when executing unmodified questions by more than 50% on both MRR and TRDR metrics. This confirms the hypothesis that using such query transformations can significantly improve search quality. However, the results of QASM and mQASM lag significantly behind the maximum achievable.

QASM outperforms Yandex on MRR and TRDR metrics but loses on the number of correct answers found. This is explained by QASM's tendency to choose too strict a transformation for a question that worked well for some questions in the training set with similar properties. And if documents are found for the transformed query, then documents containing correct answers have high rank among them (hence high MRR/TRDR). But in some cases the set of documents satisfying the too strict transformed query is empty. Yandex almost always returns a non-empty set of answers, but the desired documents don't always have high rank.

Query weights	MRR	TRDR
Equal	0.479	1.059
Selection probability	0.485(+1.2%)	1.134(+7.0%)
Selectivity estimate	0.519(+8.4%)	1.155(+9.0%)

Таблица 3: Choice of query weighting scheme for mQASM.

Without which operator?	MRR	TRDR	Answers
With all operators	0.519	1.155	33
morphological analysis cancellation	0.437 (-15.8%)	1.001 (-13.3%)	30
gluing	0.485 (-6.6%)	1.093 (-5.3%)	32
removal	0.396 (-23.7%)	0.967 (-16.2%)	32

Таблица 4: Evaluation of operator importance (mQASM)

The modified algorithm, thanks to using less strict query formulations in addition to strict ones, solves QASM's problem and demonstrates better quality compared to Yandex and QASM on all metrics.

However, mQASM's current results still significantly lag behind the maximum achievable: the algorithm managed to guess the best transformation for only 19 questions in the test set. Likely reasons for this are: irregularity of Internet data, insufficient model training (too small training question set) and insufficiently good description of queries using the set of properties used, which doesn't allow adequately learning the differences between queries.

Note also that even with full enumeration of all possible transformations and selection of the best one, the system was able to find documents containing correct answers for only 35 out of 40 questions. This fact can be explained by documents containing correct answers to unanswered questions being absent in the Yandex-indexed part of the Internet, or for some reason being assigned too low rank and no question transformation helps them appear among the first N_{suff} (in our experiments $N_{\text{suff}} = 20$) documents when Yandex ranks search results.

6.4 Query Weighting Schemes

Table 3 presents results of experiments comparing query weighting schemes for modified QASM described in section 5.5.

As expected, using equal weights leads to worse results: equal treatment of queries causes "noise"in the final set of documents when merging results of these queries.

The fact that query selectivity estimate performs better than query selection probability estimate can be explained by the fact that the most probable query is not always the best: sometimes it is too strict for the question (and then

Without which property?	MRR	TRDR	Answers
With all properties	0.519	1.155	33
number of proper names	0.420 (-19.0%)	0.886 (-23.2%)	27
number of words	0.457 (-11.9%)	1.056 (-8.5%)	32
indicators	0.445 (-14.3%)	1.025 (-11.2%)	32

Таблица 5: Evaluation of property importance (for mQASM)

no documents are found for it), sometimes too relaxed (then a large number of unsuitable documents are found). Precisely in cases when it is too relaxed, assigning it greater weight than the best query negatively affects search quality in the weighting scheme based on selection probability estimate, since many unsuitable documents can receive high weight in the final list.

6.5 Importance of Operators and Properties

To understand how the choice of sets of operators used and question properties affects the quality of results obtained, we conducted two groups of experiments.

In both cases we repeated the base experiment discussed in section 6.2, but varied the sets of operators and question properties used.

In the first case we repeated the experiment, sequentially excluding one operator from the model. As can be seen from results in table 4, removal and morphological analysis operators have the most noticeable impact on results, with cancellation of the latter affecting the number of answers found, not just their relative positions.

In the second group of experiments we sequentially excluded individual properties characterizing the question (see table 5). The most noticeable drop in effectiveness was observed when excluding information about the number of proper names.

6.6 Stability Analysis

Overall search quality results similar to those presented in table 2 depend on the question sets used for training and evaluation, and before drawing conclusions, it is important to evaluate stability of these results.

Regardless of approach, absolute search quality varies greatly depending on dataset, so averaging absolute values of search quality estimates does not allow making meaningful conclusions about stability of results. Instead, we evaluated stability of conclusions about superiority of each approach over others.

Traditionally, conclusions about superiority of search system A over system B on a given dataset are made based on measured search quality estimates for some set of information needs. [18] The difference that does not exceed a certain *significance level* is considered insignificant, i.e. neither system is considered superior, and a tie is counted.

	MRR		TRDR		Answers	
	Yandex	QASM	Yandex	QASM	Yandex	QASM
QASM	1:30:9	-	2:29:9	-	0:40:0	-
mQASM	17:3:20	36:0:4	37:0:3	40:0:0	5:1:35	40:0:0

Таблица 6: Stability of superiority conclusions (5% significance level)

In this work, stability of results was evaluated depending on the chosen partition of question set into training and evaluation sets. For this, questions in the dataset were randomly divided into training and evaluation sets in the same 60/40 ratio. A total of 40 random partitions were constructed, for each of which training and evaluation stages were fully completed.

Results of experiments are presented in table 6: each table cell contains colon-separated numbers of partitions where the method indicated in row header respectively outperformed, lost to or tied with the method indicated in column header.

As can be seen from the table, multiple QASM outperformed Yandex's result in 36 out of 40 cases on MRR estimate, and in the remaining 4 cases conclusion about superiority cannot be made since difference in results does not exceed significance level.

QASM often loses to Yandex regardless of evaluation measure chosen. Thus, we can state that despite positive effect in some cases, QASM does not give stable improvement and can lead to noticeable deterioration of results.

Multiple QASM behaves noticeably better — it always wins against original QASM. Additionally, it noticeably outperforms Yandex on TRDR, although advantage on MRR and number of answers is not as significant. In other words, multiple QASM consistently improved final ranking quality in these experiments.

Note that although such stability analysis increases validity of observations, its results may depend on model parameters (such as sets of operators or properties) and dataset used. We plan to investigate these dependencies further.

7 Conclusion

The huge volume of the Web makes it a very attractive collection for searching answers to factual queries. When processing such queries in Web context, quality of transforming natural language questions into general purpose search engine queries plays an important role.

This work investigated possibilities of using statistical approaches to transforming factual queries based on QASM algorithm and its modifications.

Conducted experimental analysis showed that application of transformations allowed by QASM model can significantly improve quality of results. Original QASM algorithm showed unstable results in our experiments, but its modification mQASM behaved much more stably.

The main goal of our research is determining and characterizing factors affecting final effectiveness of query transformation. Some results are presented in this paper, but these questions of course require further more thorough and large-scale investigation.

Список литературы

[1] E. Agichtein, S. Lawrence, and L. Gravano. Learning search engine specific query transformations for question answering. In *Proc. of the WWW-10*, pages 169–178, 2001.

[2] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. ACM Press, 1999.

[3] C.L.A. Clarke, G.V. Cormack, D.I.E. Kisman, and T.R. Lynam. Question answering by passage selection. In *Proc. of the TREC-9*, pages 673–683, 2001.

[4] C.L.A. Clarke, G.V. Cormack, M. Laszlo, T.R. Lynam, and E.L. Terra. The impact of corpus size on question answering performance. In *Proc. of the ACM SIGIR'02*, 2002.

[5] C.L.A. Clarke, G.V. Cormack, T.R. Lynam, C.M. Li, and G.L. McLearn. Web reinforced question answering. In *Proc. of the TREC-10*, pages 673–679, 2001.

[6] C.L.A. Clarke, G.V.Cormack, and T.R. Lynam. Exploiting redundancy in question answering. In *Proc. of the ACM SIGIR'01*, pages 358–365, 2001.

[7] A.P. Dempster, N.M. Laird, and D.B.Rubin. Maximum likelihood from incomplete data via the EM algorithm (with discussion). *Journal of the Royal Statistical Society series B*, 39:1–38, 1977.

[8] Susan Dumais, Michelle Banko, Eric Brill, Jimmy Lin, and Andrew Ng. Web question answering: Is more always better? In *Proc. of the ACM SIGIR'02*, pages 291–298, 2002.

[9] Eric Glover, Gary Flake, Steve Lawrence, William P. Birmingham, Andries Kruger, C. Lee Giles, and David Pennock. Improving category specific Web search by learning query modifications. In *Proc. of the SAINT'01*, pages 23–31, 2001.

[10] Sanda M. Harabagiu, Marius A. Pasca, and Steven J. Mairoano. Experiments with open-domain textual question answering. In *Proc. of the COLIN-2000*, pages 292–298, 2000.

[11] Lieming Huang, Matthias Hemmje, and Erich J. Neuhold. ADMIRE: An adaptive data model for meta search engines. In *Proc. of the WWW-9*, 2000.

[12] Cody C. T. Kwok, Oren Etzioni, and Daniel S. Weld. Scaling question answering to the Web. In *Proc. of the WWW-10*, pages 150–161, 2001.

[13] Dragomir Radev, Kelsey Libner, and Weiguo Fan. Getting answers to natural language questions on the Web. *Journal of the American Society for Information Science and Technology*, 5(53):359–364, 2002.

[14] Dragomir R. Radev, Hong Qi, Zhiping Zheng, Sasha Blair-Goldensohn, Zhu Zhang, Weiguo Fan, and John Prager. Mining the Web for answers to natural language questions. In *Proc. of the ACM CIKM 2001*, pages 143–150, 2001.

[15] M.M. Soubbotin. Patterns of potential answer expressions as clues to the right answers. In *Proc. of the TREC-10*, 2001.

[16] Ellen M. Voorhees and Dawn M. Tice. The TREC-8 question answering track evaluation. In *Proc. of the TREC-8*, pages 83–105, 2000.

[17] Zhiping Zheng. AnswerBus question answering system. In *Proc. of the HLT 2002*, 2002.

[18] И. Кураленок, И. Некрестьянов. Оценка систем текстового поиска. *Программирование*, 28(4):226–242, 2002.

[19] И. Некрестьянов, Н. Пантелеева. Системы текстового поиска для Веб. *Программирование*, 28(4):207–225, 2002.

[20] Яндекс. Вечные вопросы. *http://www.yandex.ru/skazki/skazka104.html*, Документ был доступен 01.10.2003.

Statistical Query Transformations for Question Answering in the Web

Ilya Boyandin, Igor Nekrestyanov

Abstract

We consider the problem of query transformation for question answering in the Web. The goal of such transformations is to construct a query for a traditional Web search engine given a factual natural language question so that the first several documents returned by the search engine by this query contain the correct answer to the original question.

In this paper we analyse a statistical query transformation algorithms based on the QASM [14] model, and evaluate the search quality of these algorithms on a corpus of correct Russian-language questions from the log of the Yandex search engine.