Artificial Intelligence 194 (2013) 86–110

Contents lists available at SciVerse ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

Automated query learning with Wikipedia and genetic programming

Pekka Malo *, Pyry Siitari, Ankur Sinha

Aalto University, School of Economics, P.O. Box 21210, FI-00076 Aalto, Finland

ARTICLE INFO

Article history:
Available online 19 June 2012

Keywords:
Wikipedia
Genetic programming
Concept recognition
Information filtering
Automatic indexing
Query definition

ABSTRACT

Most of the existing information retrieval systems are based on bag-of-words model and are not equipped with common world knowledge. Work has been done towards improving the efficiency of such systems by using intelligent algorithms to generate search queries, however, not much research has been done in the direction of incorporating human-and-society level knowledge in the queries. This paper is one of the first attempts where such information is incorporated into the search queries using Wikipedia semantics. The paper presents Wikipedia-based Evolutionary Semantics (Wiki-ES) framework for generating concept based queries using a set of relevance statements provided by the user. The query learning is handled by a co-evolving genetic programming procedure.

To evaluate the proposed framework, the system is compared to a bag-of-words based genetic programming framework as well as to a number of alternative document filtering techniques. The results obtained using Reuters newswire documents are encouraging. In particular, the injection of Wikipedia semantics into a GP-algorithm leads to improvement in average recall and precision, when compared to a similar system without human knowledge. A further comparison against other document filtering frameworks suggests that the proposed GP-method also performs well when compared with systems that do not rely on query-expression learning.

© 2012 Elsevier B.V. All rights reserved.

1. Introduction

A central challenge in building expert systems for information retrieval (IR) is to provide them with common world knowledge. As succinctly put by Hendler and Feigenbaum [23], in order to build any system with "significant levels of computational intelligence, we need significant bodies of knowledge in knowledge bases". That is, if a system is expected to understand the general semantics in text, closer to the way human brains do, then it should have access to the extensive background knowledge that people use while interpreting concepts (units of knowledge) and their dependencies. Of course, statistical methods and natural language processing can be used to extract semantics from text or data, but the ability of text collections to convey human- and society-level semantics is quite limited [67]. Currently, there is an ongoing quest to find new ways of integrating semantic knowledge into document modelling along with multiple other aspects (such as document timeliness and novelty) without time-consuming knowledge engineering; see e.g. Pasi et al. [48]; Meij et al. [37]; Navigli and Crisfulli [45]; and references therein. One of the emerging trends is to use socially developed resources of semantic information.

In this paper, we consider the use of Wikipedia as a source of common world knowledge for an automated query learning system. The purpose is to assist users to express their information needs as queries which are written in terms of Wikipedia's concepts instead of word tokens. The proposed system extends the Inductive Query By Example (IQBE) paradigm of

* Corresponding author.
E-mail addresses: pekka.malo@aalto.fi (P. Malo), pyry.siitari@aalto.fi (P. Siitari), ankur.sinha@aalto.fi (A. Sinha).

0004-3702/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.artint.2012.06.006

Smith and Smith [59] and Chen et al. [9] by incorporating human-level semantics using Wikipedia. The underlying principle of IQBE is quite simple: assume that a user provides a small collection of relevant (and irrelevant) example documents, the task is to learn a query based on those documents. The learnt query is then used to filter relevant documents from a newstream or document database according to the topic definition implied by the sample collection. The approach proposed in this paper uses concept-relatedness information contained in Wikipedia's link-structure to learn semantic queries using a co-evolutionary procedure. This transition from an ordinary boolean query [55] to a semantified query is necessary for integrating human- and society-level semantic information into the information retrieval (IR) system. The use of concept-based knowledge enables the IR systems to detect the relevance of a document based on the central concepts and not just words. It also allows the system to identify those documents as relevant which contain concepts closely related to the query concepts. The paper contributes towards construction of an IR framework where Wikipedia-concept based queries are learnt using a co-evolving genetic programming (GP) algorithm. The proposed framework is called Wiki-ES (Wikipedia-based Evolutionary Semantics).

The traditional automated query learning systems usually represent both queries and documents using a bag-of-words approach. Moreover, the recent studies on IQBE paradigm have almost exclusively focused on finding the best evolutionary algorithms and fitness functions for learning boolean queries; see e.g. Cordón et al. [12,13], García and Herrera [21], and López-Herrera et al. [30,29]. The use of IQBE systems is largely motivated by the portability of queries, which allows them to be interpreted as additional query generation components that can be placed on top of other retrieval systems with a boolean query interface. However, restricting the query and document models to word-level information eliminates the possibility of leveraging human-level semantics on how the different *topics* and *concepts* are related. It should be noted that a query is composed of a number of concepts, and it represents the topic the user wants to search. To illustrate the difference between word based search and concept based search, consider a situation where a user is searching for information on a particular *topic*, for which he crafts a simple query "economy AND espionage". Then, suppose that a newly arrived document has *concepts* "Trade secret" and "spying". If we now ask a human reader to judge whether the document is about economic espionage, he would most likely find it relevant due to the close relationships between the concepts. However, if only word-level information is used, the boolean query will ignore the document as the original query words never appear.

In this paper, we focus on the benefits of using concepts instead of bag-of-words in query learning and document filtering. As a test-bed for Wiki-ES system, we consider TREC-11 dataset with Reuters RCV1 corpus which provides a realistic example of a multi-domain news-stream. The experiments suggest that the concept-based approach is well-fitted to be used in conjunction with evolutionary algorithms. We observe that replacing tokens with Wikipedia's concepts yields considerable improvement in filtering results as measured by precision and recall. A comparison of Wiki-ES with other general document filtering algorithms is also drawn. The given benchmarks represent a number of paradigms. The obtained results indicated strong performance in terms of TREC-11 measures which motivates further research on the use of semantic information in document retrieval.

The structure of this paper is following. Section 2 summarizes the main contributions of the paper. Section 3 gives a review on IQBE model for automated query learning, and how Wikipedia can be used as a source of semantic information. Section 4 presents our framework Wikipedia-based Evolutionary Semantics (Wiki-ES). The co-evolutionary GP algorithm is presented in Section 5. Finally, Section 6 summarizes the experimental results.

2. Contributions

The key contributions of the paper are summarized in the following points.

2.1. Use of Wikipedia semantics in query learning

When a set of documents concerning a particular topic is to be retrieved from a database, it is common for a user to generate a query composed of tokens (terms). This query is used to decide the relevance of documents in a database by performing a search for the tokens in those documents. However, analyzing the problem from a user point of view, it is recognized that the user is not just interested in the documents containing the exact matching tokens, rather she is seeking all such documents which contain the concept represented by the token. This provides a motivation to work towards generating queries composed of concepts rather than tokens. Queries composed of concepts contain wide human- and society-level knowledge, providing a better representation of the topic being searched. In this paper, we use Wikipedia semantics to convey the concept behind a token. There is no previous study to the knowledge of the authors, which utilizes the Wikipedia semantics to construct a concept based query. The benefits of this transition from tokens to concepts, towards retrieval of documents, has been evaluated in the paper and its significance has been established.

2.2. Development of a co-evolving GP

Generating an accurate query for a search is often an iterative and tedious task to perform. However, if there is a set of documents available at hand, with each document marked relevant or irrelevant, the task of query generation can be entirely avoided by directing the documents to a genetic programming algorithm. Based on the relevance or irrelevance of

the training documents, a concept based query can be learnt by the algorithm, saving the user from a monotonous task. The paper contributes towards development of a co-evolving evolutionary algorithm specialized to generate concept based queries for document retrieval. The algorithm takes a set of training documents as input. Each document in the training set is marked as relevant or irrelevant by the user, based on which the algorithm produces concept based queries. The outcome of the algorithm is not a single query, rather a set of queries which are put together using a voting function. The use of multiple queries and a voting function leads to avoidance of any over-fit to the training set which may happen if only a single query is generated. Multiple queries produced by the algorithm occupy different high fitness niches in the objective space and contribute towards the final decision for a document being relevant or irrelevant. Though genetic programming has been widely used for query construction, the implementations have relied on token-based queries [13,30,29].

2.3. Comparison with general document filtering methods

The paper performs a broad comparison with the existing methodologies representing a number of different paradigms. All approaches have been evaluated using TREC-11 relevance statements on hundred different topics. The evaluation consists of three experiments. In the first experiment, the performance of Wiki-ES is evaluated against state-of-the-art token-based genetic programming procedure. The second experiment compares Wiki-ES with the well-known classification algorithms, SVM and C4.5. Both concept-based and token-based profiles are considered. Finally, in the third experiment, a comparison with other contemporary approaches is drawn.

3. Prelude: Wikipedia semantics and IQBE

To provide an idea on the wealth of Wikipedia's semantic information and how that information can be utilized in query learning, we briefly discuss the recent innovations which leverage Wikipedia's link-structure to produce low-cost measures on the relationship between concepts and topics. In this section, we also summarize the recent developments in automated query learning. In particular, we consider the work inspired by evolution-based genetic algorithms, and the IQBE paradigm of Smith and Smith [59], Chen et al. [9], Cordón et al. [13], and López-Herrera et al. [30,29].

3.1. Wikipedia as a semantic knowledge-resource

Research on ontology-based knowledge models has been largely motivated by their ability to provide unique definitions for concepts, their relationships and properties, which together create a unified description of a given domain. Having access to such structured information in machine-readable form has provided standardized ways for sharing common knowledge and, thus, enabled its efficient reuse in applications. Despite these advantages, the use of ontologies has been limited because of the large engineering costs that are unavoidable in manually built knowledge-resources. Furthermore, there is the difficulty of keeping the resources updated, in particular, when multiple domains are considered. As it is commonly known [60,34], even the most extensive ontologies, such as the Cyc ontology, have limited and patchy coverage. Therefore, the urgent need to find less expensive ways to describe concepts and their dependencies is well recognized. This has motivated research towards the use of socially or automatically constructed knowledge-resources.

When speaking of readily accessible multi-domain knowledge resources, the one that instantly comes into the mind is Wikipedia. Thanks to the activity of numerous volunteers, Wikipedia has rapidly matured into one of the largest repositories of manually maintained knowledge. Today, there are already over 3.3 million articles in English Wikipedia, and more arrive on a daily basis. The popularity of Wikipedia has also stimulated increasing research to investigate how the semantic information in Wikipedia can be harnessed for a variety of uses; see Medelyan et al. [34] for a comprehensive review. We acknowledge the seminal work done by Ponzetto and Strube [51,53,52,61], Gabrilovich and Markovitch [18–20], Milne et al. [39,42,40,41], Medelyan et al. [35,33], Nastase et al. [43,44], and Mihalcea and Csomai [38], who have examined different ways of using Wikipedia to perform automated concept-recognition and compute semantic relatedness between pairs of concepts. Considerable efforts have been also done to integrate Wikipedia with WordNet to create large multilingual semantic networks; see e.g. Navigli and Ponzetto [46]; and Ponzetto and Navigli [50]. However, there is relatively little research on the use of concept-based knowledge in document retrieval. To our knowledge, the only paper discussing this topic is by Egozi et al. [14] where explicit semantic analysis (ESA) technique of Gabrilovich and Markovitch [18] and pseudo-relevance feedback are used for document ranking.

3.1.1. Wikipedia-concept

In spite of the fact that Wikipedia does not really fulfill the criteria of being an ontology, a closer look at its structure reveals many similarities [24]. By interpreting Wikipedia's articles as concepts, and by regarding the overall link-structure – including redirects, hyper-links, and category links – as relations, it is warranted to argue that Wikipedia is the largest semantic network available today. As nicely captured by Medelyan et al. [34], Wikipedia provides a solid middle-ground between ontologies and classical thesauri "by offering a rare mix of scale and structure". Indeed, the recent developments suggest a number of ways in which Wikipedia can be used for extracting ontological knowledge; for example, see the Yago-ontology of Suchanek et al. [62] and WikiNet by Nastase et al. [44].

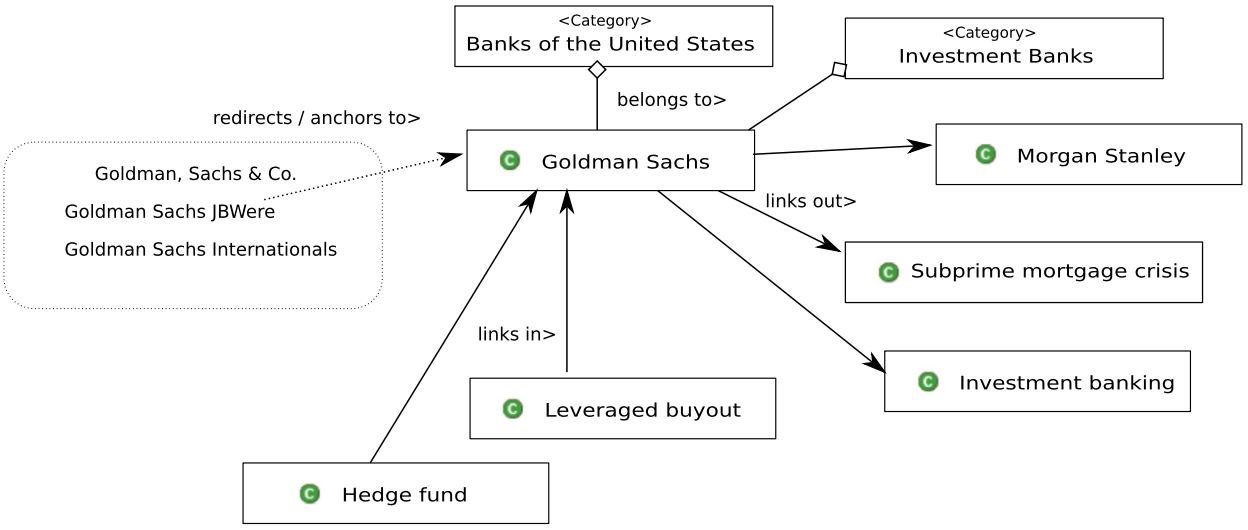

Fig. 1. "Goldman Sachs" as a Wikipedia-concept.

The primary feature that makes Wikipedia considerably richer in semantic knowledge than a conventional thesaurus is its dense internal link-structure. To illustrate the notion of Wikipedia-concept a bit more closely, let us consider, for instance, Wikipedia's article on "Goldman Sachs" (Fig. 1). Each Wikipedia-concept (article) belongs to at least one or more categories, which provide information about broader topics, hyponyms and holonyms. In this case, we find that Goldman Sachs belongs to categories such as "Investment Banks" and "Banks of the United States". Moreover, if the article's topic is sufficiently broad, then there also exists an equivalent category with the same title as the article. In addition to category-relationships, the articles have lots of hyper-links that represent semantic relationships between concepts. On average, each article refers to about 25 other articles. For instance, "Goldman Sachs" has links to many other banks (e.g. "Morgan Stanley") and financial concepts (e.g. "Subprime mortgage crisis). These linkages can be exploited in various ways to mine knowledge on concepts and their relationships. Finally, to account for synonyms and alternative spellings of the article's name, each article has also a number of redirects that connect to the article. The redirects are complemented by anchors, which represent the words used within hyper-links that refer to the given article; and when several articles could be given the same name (e.g. Bank), then there is a disambiguation page that lists the alternative senses corresponding to that name.

Therefore, considering the wealth of semantic information conveyed by Wikipedia, we find it natural to treat the Wikipedia-articles as equivalents for ontological concepts when modelling documents and queries. To formalize these ideas, we employ the following notation while referring to Wikipedia-concepts:

Definition 3.1.1 *(Wikipedia-concept).* Let W denote the collection of Wikipedia-articles available for language Σ. Then a Wikipedia-concept is defined as an article $w \in W$, which is a uniquely identified representative of a certain concept.

Once we have the definition, there at least two questions that follow. The first one is concerned with concept-recognition. Clearly, it is not uncommon to find that several concepts may share the same textual representation. Thus, being able to resolve whether a certain concept is present in a document or not is a non-trivial problem. In the literature, this is commonly referred to as the wikification task [38] or automatic topic-linking problem [40,35]. This will be discussed more closely when outlining the content model used by Wiki-ES; see Section 4.2.

The second question, discussed in the following Section 3.1.2, concerns the way semantic relatedness between any (concept, concept)-pair and (concept, document)-pair is measured. This needs to be resolved before we discuss the idea behind Wikipedia-based query rules and the way they are learnt from example documents provided by a user. In particular, we need the notion of semantic relatedness while evaluating whether a document matches the given query or not.

3.1.2. Measuring semantic relatedness

Although approaches to measuring conceptual relatedness based on corpora or WordNet have been around for quite long (McHale [32] and Finkelstein et al. [16]), the use of Wikipedia as a source of background knowledge is a relatively new idea. The first step in this direction was taken by Strube and Ponzetto [61], who proposed their WikiRelate-technique that modified existing measures to better work with Wikipedia. This was soon followed by the paper of Gabrilovich and Markovitch [19], who suggested explicit semantic analysis (ESA) to define a highly accurate similarity measure using the full text of all Wikipedia articles.

The most recent proposal is, however, the Wikipedia Link-based Measure proposed by Milne et al. [39,40], where only the internal link structure of Wikipedia is used to define relatedness. The approach is known to be computationally very

cheap and has still achieved relatively high correlation with humans, which is why we have adopted it as a basis for the similarity measures used in this paper. The relatedness measure essentially corresponds to the Normalized Google Distance inspired by Cilibrasi and Vitanyi [10].

Definition 3.1.2 *(Link-relatedness).* (See Milne et al. [39,40].) Let w_1 and w_2 be an arbitrary pair of Wikipedia-concepts, and let $W_1, W_2 \subset W$ denote the sets of all articles that link to w_1 and w_2, respectively. The link structure-based concept-relatedness measure, link-rel : $W \times W \to [0,1]$, is then given by

$$\text{link-rel}(w_1, w_2) = \begin{cases} 1 - ND(w_1, w_2) & \text{if } ND(w_1, w_2) \leqslant 1, \\ 0 & \text{otherwise} \end{cases}$$

where

$$ND(w_1, w_2) = \frac{\log(\max|W_1|, |W_2|) - \log(|W_1 \cap W_2|)}{\log(|W|) - \log(\min(|W_1|, |W_2|))}$$

denotes the normalized Google-distance.

Remark 3.1.3. Although, this link-based relatedness measure is defined only for uniquely identified Wikipedia-concepts, it can be extended for calculating relatedness between any given pair of n-grams by using our knowledge about redirects and anchors attached to different concepts, see discussion in Malo et al. [31].

The underlying principle of link-rel is rather simple: if two articles share a lot of same links, then they are likely to be highly related. For example, if we consider two major investment banks, such as "Goldman Sachs" and "Morgan Stanley", the link-rel yields a relatedness score of almost 80 percent due to the large number of financial concepts shared by both bank-articles. Whereas "Goldman Sachs" and "Football" are 0 percent related. Of course, these results are sensitive to the quality of the concept-articles' link-structure, and can thereby vary depending on the version of the Wikipedia being used. Nevertheless, when well-established articles are considered, and when speed is essential, we find that this kind of graph-based approach has proven to be a reasonably reliable way of measuring proximity between any arbitrary pair of concepts.

So far, we have considered the computation of semantic relatedness in its conventional setup between two concepts. However, given our intention to use Wikipedia's relatedness information in matching queries and documents, it is perhaps more relevant to ask: how can we measure the relatedness between a document and a given concept? Or how likely is it for the given concept to appear in the document? For this purpose, we propose the following simple extension of the link-relatedness measure.

Definition 3.1.4 *(Document-concept relatedness).* Let $w \in W$ denote any Wikipedia-concept, and $d \in \mathcal{D}$ be an active document. The Wikipedia-based document-term-relatedness measure, d-rel : $W \times \mathcal{D} \to [0,1]$, is given by

$$\text{d-rel}(w, d) = \max\{\text{link-rel}(w, \bar{w}):\ \bar{w} \in \Lambda(d)\}$$

where the document model, $\Lambda(d)$, is interpreted as the collection of Wikipedia-concepts detected in document d; see Section 4.2 for further discussion on document modelling.

Here, the use of maximum rather than sum-based operator such as average is a deliberate choice. Since d-rel is intended to be used in evaluating whether a document matches a given query, we do not want to allow any sum-operations to mask the presence of those concepts in a document which are not related to its central theme. To illustrate the idea, consider a single-concept query for documents on "Industrial espionage". Now, suppose that we receive a large document on car manufacturing, where most of the discussion is concerned with general economics and car models. However, the document still has a single paragraph on stolen trade secrets and car-prototype specifications. In order to prevent the document's main theme from hiding its relatedness to industrial spying, we choose to measure the relatedness by using the concept that is best associated with espionage. In this particular case, because trade secret is strongly linked to industrial espionage, it is natural to use their association to evaluate the overall relatedness between the document and the given query.

3.2. Query learning problem

The demand for automated query learning is driven by the difficulty of formulating effective queries that match the user's information needs. Finding appropriate search terms and conditions is generally hard even for expert users. Therefore, given a certain topic, the task of query learning systems is to help the user to find a query definition with improved precision and recall. As the size of world's information base is growing at a staggering rate, the problem is becoming increasingly pressing. To alleviate it, a large number of competing solutions for query formulation have been proposed in response. As suggested by Cordón et al. [12], these can be divided into three categories: (1) term learning; (2) weight learning; and (3) query-structure learning.

The commonality of the approaches is their reliance on some form of relevance feedback, where the system elicits (possibly iteratively) a set of feedback statements from the user. In the first two model categories, relevance feedback is used for modifying the user's previous query by removing or adding terms and adjusting their weights to better reflect the user's relevance judgements. For example, many of the probabilistic models and document-vector modification models belong to these categories; see e.g. Salton and Buckley [58] and Rocchio [57], Yang and Korfhage [66], Horng and Yeh [25], and Boughanem et al. [3,4].

Our focus is on the third category, query-structure learning, which takes the learning process one step further in the context of boolean or fuzzy boolean queries. It not only attempts to infer the terms that are most appropriate for representing a given query but also tries to learn the query's structure, i.e. it determines how the boolean operators AND ($\wedge$), OR ($\vee$), and NOT ($\neg$) should be used to join the different concepts. In many texts, *query learning* is considered as a reserved word for representing this third type of query definition, where both the functional form and query-terms are free variables; see e.g. Cordón et al. [12,13], López-Herrera et al. [30,29] and their references. The IQBE paradigm (Section 3.3) and the Wiki-ES system introduced in this paper are mainly viewed as structural query learning models. In the context of this paper, where each query-term represents a concept, we define the query learning problem as follows.

Definition 3.2.1 *(Query learning problem).* Let C be a set of admissible concepts, and let Q denote the space of all admissible queries which can be formed using concepts in C. The query learning task is to find that boolean expression from the set Q which best represents the user's information needs by applying the following syntactic rules:

1. Atomic query (single concept): $\forall q = c_i \in C \rightarrow q \in Q$.
2. Composition using AND: $\forall q, p \in Q \rightarrow q \wedge p \in Q$.
3. Composition using OR: $\forall q, p \in Q \rightarrow q \vee p \in Q$.
4. Negation: $\forall q \in Q \rightarrow \neg q \in Q$.

The space of admissible queries Q consists of all the queries obtained by applying the above set of rules.

There are many ways to approach the above problem – both with and without the use of semantic knowledge. At this stage, we notice that the definition remains deliberately abstract by not specifying how the set of concepts should be understood and how the learnt queries be matched against documents. Of course, when classical boolean queries using the bag-of-words approach are considered, the answer is quite straightforward. However, when the atoms of a query are uniquely defined concepts, it is no longer self-evident how the query should be evaluated. In fact, as we find out in Wiki-ES model, the performance differences between concept-based and word-based approaches follow from the way concept-relationship information is incorporated into the learnt queries.

3.3. IQBE – inductive query by example

One of the best known bag-of-words based methods for solving the query learning problem 3.2.1 is the Inductive Query By Example (IQBE) framework originated by Smith and Smith [59] and Chen et al. [9]. The idea behind IQBE paradigm is in principle very similar to relevance feedback; both of them require explicit relevance statements from the user to guide the retrieval process. In IQBE, the user provides the system with a collection of sample documents (positive/negative examples) from which an algorithm learns the terms and the boolean operators joining them, such that the obtained query best represent the user's information need. However, instead of modifying an existing query iteratively, the system performs only a single run to generate a fresh query from scratch. Once the learnt query is available, it can be executed on any information retrieval system (IRS) that accepts boolean queries. Such portability of queries can be considered as one of the advantages that distinguishes IQBE systems from general relevance feedback. In descriptions of IQBE architecture, this is commonly emphasized by presenting IQBE system as a separate unit outside the IRS; see López-Herrera et al. [30] and Fig. 2 for descriptions of a general IQBE system.

In IQBE framework, the query learning task is viewed as a large optimization problem, where the search space consists of all possible queries that can be presented to the IRS. Therefore, recognizing the high dimensionality, discreteness and non-linearity of this problem, it is no surprise that the IQBE approaches usually rely on some form of evolutionary computation. In particular, following the early studies by Kraft et al. [27] and Smith and Smith [59], genetic programming [26] has gained ground as a robust choice for query learning. Recently, a number of frameworks based on multi-objective genetic programming have also been examined. Due to the fact that the performance of an IRS is mostly evaluated in terms of precision and recall, it appears natural to consider query learning as an inherently multi-objective problem. Consequently, most of the recently introduced IQBE frameworks are utilizing various combinations of genetic programming and multi-objective evolutionary algorithms to learn Pareto-frontiers of optimal query expressions, see e.g. Cordón et al. [13] and López-Herrera et al. [30,29].

As discussed by Tamine et al. [63], the popularity of evolutionary algorithms is largely explained by their diversity which allows them to search different regions of the solution space simultaneously. It is also argued that evolutionary algorithms are less sensitive to the quality of the initial query. Whereas classical relevance feedback methods, such as Rocchio [57], perform poorly if the initial query fails to retrieve relevant documents. The probabilistic exploration induced

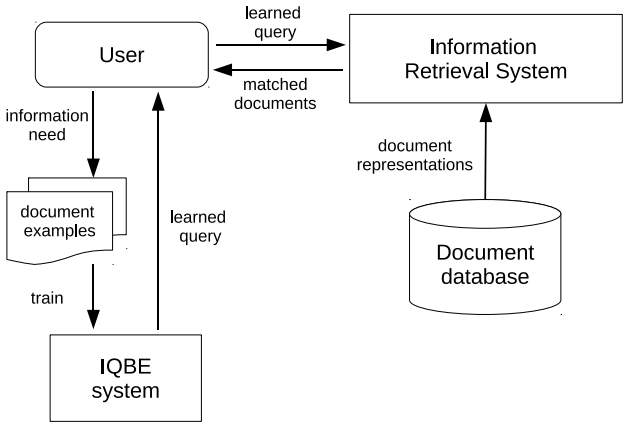

Fig. 2. A general IQBE architecture.

by evolutionary algorithms permits them to search unexplored areas independent of the initial query [7]. Hence, the use of evolutionary algorithms is a well-justified choice for query learning as non-expert users can rarely find a good query on a first try when complicated topics are considered.

Although the automated query learning problem has stimulated a lot of interest over the past few years, it is noteworthy that majority of the development has concentrated on improving learning algorithms rather than coming up with ways to enrich the query with semantic information. However, recognizing the fact that the use of semantic information has transformed many natural language processing applications [67], we consider it worthwhile to work towards the development of a Wikipedia-concept based approach which would enhance automated query learning.

4. Wiki-ES: Learning semantic queries with Wikipedia

In this section, we present the Wiki-ES (Wikipedia-based Evolutionary Semantics) framework for automated query learning. The approach is based on the Genetic Programming (GP) paradigm, which is a potent tool in artificial intelligence for performing program induction. In GP, the idea is to use the principles of evolutionary computation to intelligently search the space of possible computer programs for finding an individual that is highly fit for solving the problem at hand. In effect, one could say that the purpose is to get the machine to generate a solution to the problem without being explicitly programmed [26]. For example, in our case we want the Wiki-ES system to learn a program (i.e. query) that leads to recovery of a high number of relevant documents while keeping the irrelevant documents aloof. The learning process is driven by the evolutionary pressure which guarantees that only the fittest individuals among all potential query candidates survive.

4.1. Wiki-ES framework overview

A bird-eye's view of the Wiki-ES framework resembles the architecture of the IQBE paradigm (see Fig. 2), where the idea is that the system is able to learn an optimal query by using just a small set of sample documents that represent the user's current topic or information need. On the surface, this sounds simple. However, when examining the steps involved in the learning process, it becomes clear that a number of choices, ranging from the choice of query and document models to the choice of the genetic procedure, have large impacts on the outcome.

To illustrate the way Wiki-ES approaches the query learning problem, let us consider an example where a user seeks to define a query that picks up all the documents on economic espionage but ignores the ones on politics or military espionage. Then, we can split the Wiki-ES process into the following steps (see Fig. 3):

1. *Training data generation*: Suppose that the user has already found a collection of documents that she considers highly relevant for the topic and also a collection of documents that are concerned with espionage but are more about military spying than industrial espionage. Then, the training data set is defined as a relevance matrix, where each sample document is given a boolean value to represent its relevance for the topic ($1 =$ relevant, $0 =$ irrelevant).
2. *Learning a query-expression*, i.e. the Wiki-ES rule: In the learning step, the training data set is given to the GP-algorithm to find an optimal Wiki-ES rule to describe the topic. Each Wiki-ES rule is a weighted sum of a number of queries, which decides for the relevance of a given document. A detailed description of the rules is given in Section 4.3. The GP-algorithm is described in Section 5.
3. *Feeding the Wiki-ES rule and documents to the Wiki-based Information Retrieval System (WIRS)*: Once the optimal Wiki-ES rule is known, it can be given to a matching subsystem which evaluates the query against the incoming documents. In Wiki-ES framework, this task is handled by WIRS module, which consists of two subsystems: the document modeling subsystem and the rule-matching subsystem.

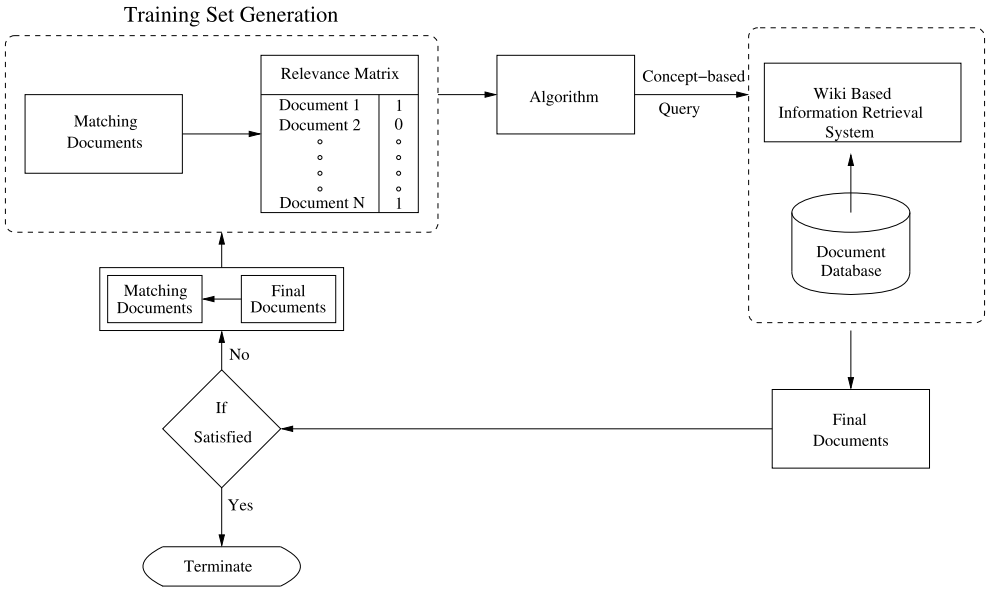

Fig. 3. Wiki-ES flowchart.

(a) *Document modeling subsystem*: Before the incoming documents can be matched against the Wiki-ES rules, they are passed through a wikifier and a named-entity recognizer (NER). The resulting profile, expressed in terms of the identified Wikipedia concepts and named-entities, can then be used to represent the document contents when matching against Wiki-ES rules; see Section 4.2 for description of the document model.

(b) *Rule-matching subsystem*: The rule-evaluator in WIRS module provides a matching subsystem for deciding whether a given document matches the currently active semantic rule or not. In Wiki-ES framework, it is hence the responsibility of the rule-evaluator to utilize Wikipedia's concept-relatedness information while determining whether the query concepts are present in the active document – either directly or indirectly. The way how the rule-evaluator operates is described in Section 4.3.

4. *Returning the filtered documents to user*: The documents that are found to match the active Wiki-ES rule, are returned to the user and the others are discarded.

4.2. Wikipedia-based document model

In Wiki-ES framework each document is represented by a collection of Wikipedia-concepts that are identified from its contents. The approach builds on the wikification technique proposed by Milne et al. [40] and Medelyan et al. [35], where a two-stage classifier is utilized to recognize those terms in the document which should act as Wikipedia-concepts. The model employed in this paper extends the wikification-process by splitting the found concepts into two categories, general Wikipedia-concepts and named-entity concepts, using a named-entity recognizer.

To explain the rationale for this modification, consider, for example, a named entity "Goldman Sachs" and a general concept "Investment banking". Now, to say that a certain document discusses Goldman Sachs requires that the bank's name is explicitly mentioned. On the other hand, if we say that a document is about investment banking, it is sufficient to find a collection of investment banking related concepts rather than the exact concept name to identify the document as relevant. Clearly, the different nature of general concepts and named-entities should be taken into account when specifying the sensitivity of the Wiki-ES model to different concept types. Hence, in Wiki-ES, each document is interpreted as a pair of two collections: the named-entities and other Wikipedia concepts.

Definition 4.2.1 *(Wiki-ES document model).* Let D be the space of documents, and W denote the collection of Wikipedia-concepts. The document model $\Lambda(d)$ is defined as a subset of Wikipedia-concepts given by mapping,

$$\Lambda : d \in D \mapsto N_d \cup G_d \subset W,$$

where N_d and G_d denote the sets of named-entities and general Wikipedia-concepts found in the document d.

Example 4.2.2. If a document $d \in D$ contains Wikipedia-concepts, e.g. {Investment banking, Goldman Sachs, Morgan Stanley, Mortgage, Credit}, we simply present the document model as a union of two separate collections, $\Lambda(d) = N_d \cup G_d$, where $N_d = \{\text{Goldman Sachs, Morgan Stanley}\}$, $G_d = \{\text{Investment banking, Mortgage, Credit}\}$.

The document model Λ is implemented in two parts: wikification and named-entity recognition.

4.2.1. Wikification

The concept-identification technique used in this paper is based on the wikification (or topic-indexing) algorithm proposed by Milne et al. [40]. The algorithm is implemented as a two-step classification task:

(i) Disambiguation-step: The wikification process begins with a search for link candidates in a document. After the link candidates have been found, the problem is to associate them with the correct concepts (Wikipedia articles). The sense with maximum semantic relatedness to the document is selected. The classifier responsible for the task is trained on a collection of features which describe the prior probabilities of alternative senses, their average semantic relatedness to the other context terms, and the quality of the context.
(ii) Detection-step: The next stage is to decide which of the concept links should be retained in the final profile. To ensure that all central concepts get linked and the unnecessary ones are eliminated, the classifier is trained using average relatedness information of each concept and the other candidates. Other features include link probability, disambiguation confidence (probability given by the first classifier), generality (the depth of the concept in Wikipedia's category tree), and location and spread (the distance between first and last occurrence).

Both classifiers are implemented using Quinlan's [54] C4.5 algorithm.

4.2.2. Named-entity recognition

The second part of the document profiling, named-entity recognition (NER), is done by using the Conditional Random Fields (CRF)-based classifier proposed by Finkel et al. [15]. The CRF-framework is an undirected graphical model, which defines a single log-linear distribution over the named-entity labels conditioned on the observation sequence; see e.g. Lafferty et al. [28] for general discussion. The NER-algorithm of Finkel et al. [15] is a modified CRF, where Gibbs sampling techniques are utilized to enable efficient inference with non-local structures. By relaxing the requirement of exact inference, the framework permits the use of long-distance dependency information, enforcement of label consistency and extraction of template constraints. Due to the fact that natural language contains a lot of non-local structure, the proposed model is particularly suitable for named-entity recognition tasks.

Once the document has been wikified, the named-entity recognition task is carried out in two steps. First, we execute the CRF-model independently to find out the terms that can be interpreted as named-entities. The CRF-model used in this paper has been trained on CoNLL 2003 dataset by using a collection of features described by Finkel et al. [15]. Although the model provides information on the classes of the named-entities (i.e. whether they are people, places or organizations), we do not utilize this classification but instead all named-entities regardless of their types are combined into a single annotation set. The Wikipedia-concepts identified as named-entities are collected in the set N_d, and the remaining general Wikipedia-concepts are collected into the set G_d. The named-entities identified by the CRF-classifier which are not Wikipedia-concepts are discarded. Concerning future development, it is noteworthy that the efficiency of the named-entity recognition step can be greatly enhanced by combining it with the wikification stage. Another way to approach this would be to classify the articles against entity types before-hand; see Nothman et al. [47].

4.3. Query model: structure and matching of Wiki-ES rules

As mentioned in Section 4.1, each Wiki-ES rule can be viewed as a composition of a number of queries. The Wiki-ES rule has an underlying structure that is essentially different from what is seen in ordinary boolean queries. To provide a more accurate picture, we formalize the definition of Wiki-ES rule as a voting system where several concept based queries go for a voting and the weighted sum of their votes is taken to represent the relevance of a document.

The presentation of the Wiki-ES model is structured as follows. First, we define the Wiki-queries that are used as building blocks in Wiki-ES rule (Section 4.3.1). Thereafter, in Section 4.3.2, we introduce a fitness-measure for evaluating the quality of individual queries, and discuss how a voting system can be used to combine the output of several Wiki-queries to generate a Wiki-ES rule. Section 4.4 summarizes the Wiki-ES learning problem. We also discuss the benefits of constructing the Wiki-ES rule as a voting system instead of using the individual queries directly.

4.3.1. Building blocks of Wiki-ES rules

Now, we begin by outlining the types of boolean queries used as building blocks for the Wiki-ES rule. To distinguish these from ordinary term-based queries, we refer to them as Wiki-queries (concept-based queries) hereafter. Unlike an ordinary boolean query, a Wiki-query consists of two parts. In addition to the query-expression, each Wiki-query also contains a specialized evaluator function which allows the query to utilize Wikipedia's concept-relatedness information when it is matched against documents.

Definition 4.3.1 *(Wiki-query).* A Wiki-query $q : D \to \{0, 1\}$ is defined by a pair (e, δ), where

(i) the first component, e, is an ordinary query-expression that is defined in terms of Wikipedia-concepts $V \subset W$ and the standard boolean operators by following the syntactic rules outlined in 3.2.1; and
(ii) the second component, $\delta : V \times D \to \{0, 1\}$, is a concept-evaluator function given by 4.3.2, which determines whether a concept $v \in V$ is present in any given document $d \in D$.

When matching the given query $q = (e, \delta)$ against any document d, the value of the query $q(d)$ is obtained by replacing each concept $v \in V$ in the query expression e with the corresponding value $\delta(v, d)$ given by the concept-evaluator.

Definition 4.3.2 *(Concept-evaluator).* The concept-evaluator function, $\delta : V \times D \to \{0, 1\}$, whose purpose is to account for Wikipedia's concept-relatedness information when evaluating Wiki-queries, is given by

$$\delta(v, d) = \begin{cases} 1 & \text{if } v \in \Lambda(d), \\ 1 & \text{if } v \in \mathrm{Rel}(d), \\ 0 & \text{otherwise} \end{cases}$$

where

$$\mathrm{Rel}(d) = \big\{ v \in V\colon \text{d-rel}(v, d) > c_{\mathrm{rel}}(v) \big\},$$

and $c_{\mathrm{rel}} > 0$ is a threshold function controlling the acceptance sensitivity by relatedness criteria. The threshold for document-concept relatedness function (d-rel) depends on the type of concept, i.e. whether it is a named-entity or general Wikipedia-article. If $\Lambda(d) = N_d \cup G_d$, we have

$$c_{\mathrm{rel}}(v) = \begin{cases} c_1 & \text{if } v \in N_d, \\ c_2 & \text{if } v \in G_d. \end{cases}$$

Each sensitivity threshold is chosen based on training data. In the preliminary experiments carried out in this paper, relatively high threshold values $c_1 \approx 0.95$ and $c_2 \approx 0.70$ were chosen to maintain precision. The purpose of the distinction between named-entities and general concepts is to allow stricter thresholds for named-entities which have narrower definitions than general concepts.

Example 4.3.3. Let q be defined by (e, δ). If $e = v_1 r_1 v_2 r_2 \cdots r_{k-1} v_k$, and $v_i \in W$, $r_i \in \{\wedge, \vee, \neg\}$ for all $i = 1, \ldots, k$, then the value of the query amounts to $q(d) = \delta(v_1, d) r_1 \delta(v_2, d) r_2 \cdots r_{k-1} \delta(v_k, d)$.

To illustrate the underlying idea, consider a simple Wiki-query, $q = (e, \delta)$, where the query-expression

$$e = \mathit{Lawsuit} \wedge (\mathit{Espionage} \vee \mathit{TradeSecret}) \wedge \mathit{BMW}$$

requests for documents on industrial espionage that are concerned with BMW. Now, suppose that the following document d is received, then the first step is to perform the profiling:

> A civil court in Hamburg will give its verdict on Tuesday on a hearing called by Spiegel, a leading German magazine. Spiegel is trying to lift an injunction from VW preventing it from repeating allegations of corporate spying against Mr Lopez... The documents include top-secret details of Opel's new small car project, coded the O-car, which is to rival Volkswagen's planned Chico.

$G_d = \{\text{Lawsuit, Allegation, Automobile, City car, Corporation, Injunction, Classified information, Project}\}$,

$N_d = \{\text{Hamburg, Der Spiegel, Opel, Volkswagen}\}$,

Non-Wiki NERs: {Mr Lopez, O-car, Chico}.

After the document has been profiled, it can be evaluated against the query expression. In this case, during the concept-evaluation step, we find that $\delta(\mathit{Lawsuit}, d) = 1$ because the terms "civil court" and "allegation" point to *Lawsuit*, and similarly we have $\delta(\mathit{Espionage}, d) = 1$ because "spying" is a redirect to *Espionage*. However, the evaluation of the concept *TradeSecret* and the named-entity concept *BMW* turn out to be more problematic as they will depend on the acceptance-sensitivity function (c_{rel}).

Let us first consider the *TradeSecret*-concept. To determine whether *TradeSecret* is present in the document, we need to examine its relatedness to other concepts that have been identified from the document. In the above excerpt "top-secret" is recognized as *ClassifiedInformation* which is strongly related to *TradeSecret*, therefore the decision boils down to the comparison of these two concepts. Here, $\delta(\mathit{TradeSecret}, d)$ equals 1 only if the acceptance sensitivity $c_{\mathrm{rel}}(\mathit{TradeSecret})$ is less than the link-relatedness measure between *TradeSecret* and *ClassifiedInformation*.

So far, it seems that the given excerpt is almost a match provided that the last concept, *BMW*, is also recognized as related to the document. Now, the acceptance sensitivity parameter for named-entities c_1 is set at a reasonably strict-level, say 0.95, to ensure that named-entities are not as broadly defined as the general concepts. For example, one would observe a very high relatedness between *BMW* and *VW* as they are both German car manufacturers with almost similar link-structures. However, mixing these two would be a serious error from the user's point of view. Therefore, being able to define acceptance sensitivities separately for named-entities and general Wikipedia-concepts proves to be a useful tool. Eventually, due to high value of c_1, we deduce that $\delta(BMW, d) = 0$, and therefore the document is considered to be irrelevant.

4.3.2. Wiki-ES rule

Having introduced Wiki-queries, we are now ready to explain how they are combined to generate a Wiki-ES rule. For this purpose, we define two additional functions: (i) a fitness-function for measuring the quality of individual Wiki-queries; and (ii) a voting function for summarizing the output of a group of Wiki-queries into a single measure.

Definition 4.3.4 *(Fitness of Wiki-query).* Let Q denote the space of admissible Wiki-queries. The fitness-function for a Wiki-query $q \in Q$ is defined as the mapping, $F : (q, D_t) \mapsto c \in [0, 1]$, which corresponds to the F-score within a given set of training documents $D_t \subset D$:

$$F(q, D_t) = \frac{2P(q, D_t)R(q, D_t)}{P(q, D_t) + R(q, D_t)},$$

where $P(q, D_t)$ is the precision of the query in the document set D_t, and $R(q, D_t)$ is the recall of the query, respectively. By denoting the relevance of a document $d \in D_t$ by $r(d) \in \{0, 1\}$, precision and recall are defined as

$$P(q, D_t) = \frac{\sum_{d \in D_t} r(d)q(d)}{\sum_{d \in D} q(d)} \quad \text{and} \quad R(q, D_t) = \frac{\sum_{d \in D_t} r(d)q(d)}{\sum_{d \in D} r(d)}.$$

Now, suppose that instead of having a single query to describe the user's information need, we have several complementary queries for the same topic, where each query represents a part of the user's need. In order to benefit from the diversity provided by the multiple query representation, we first need to resolve how the potentially conflicting results from different queries can be combined into a single document-relevance measure. Given the above F-score as a fitness-measure for evaluating the quality of each individual Wiki-query, a natural approach for dealing with this "query fusion" problem is to consider the following voting function where each query contributes to the overall relevance judgement according to its relative fitness:

Definition 4.3.5 *(Voting function).* Let $A \subset Q$ be a finite collection of Wiki-queries. A voting function $\mu_A : D \to [0, 1]$ is given by

$$\mu_A(d) = \frac{\sum_{i=1}^{|A|} F_i q_i(d)}{\sum_{i=1}^{|A|} F_i},$$

where $F_i = F(q_i, D_t)$ is the fitness of query q_i evaluated with respect to a training document set $D_t \subset D$.

Remark 4.3.6. The voting function μ_A can be also used for ranking the documents based on their relevance to the given topic. However, the use of rank-order information is left as a direction for further research.

The value of the voting function has an interpretation as the joint-relevance of a document, where the judgement is based on several alternative queries that describe the given topic. If the value of the voting function is greater than 0.5, then the document is considered relevant, otherwise it is considered irrelevant. Using this weighted contribution, the information from several queries is taken into account, which helps to reduce the risk of over-fitting the training document set with a single query. This discussion is formalized by the following definition of the Wiki-ES rule.

Definition 4.3.7 *(Wiki-ES rule).* Let Q denote the space of admissible boolean queries formed using Wikipedia-concepts, and let μ_A be a voting function that evaluates the document-relevance based on a finite set of Wiki-queries, $A \subset Q$. Now, the Wiki-ES rule is defined as the function $\bar{q}_A : D \to \{0, 1\}$:

$$\bar{q}_A(d) = \begin{cases} 1 & \text{if } \mu_A(d) > 0.5, \\ 0 & \text{otherwise} \end{cases}$$

and the space of admissible Wiki-ES rules is given by $\bar{Q} = \{\bar{q}_A \mid A \subset Q\}$, where A denotes any finite set of Wiki-queries.

Remark 4.3.8. At this point, it is worthwhile to note that any Wiki-query can be viewed as a Wiki-ES rule, i.e. $Q \subset \bar{Q}$, because for every Wiki-query $q_0 \in Q$, we have $\bar{q}_{\{q_0\}} \in \bar{Q}$. Hence, the Wiki-ES rules provide a natural extension of the Wiki-queries.

Table 1
The interpretation of GP-components in Wiki-query context.

GP component	Meaning in Wiki-query
Terminals (leaf nodes)	Wikipedia-concepts in a query-tree
Functions (non-leaf nodes)	Boolean query operators (AND, OR, NOT)
Fitness function	The objective function (F-score) in the query learning problem
Reproduction, crossover, and mutation	Genetic operators for driving the development of Wiki-queries according to the evolutionary principles

4.4. Wiki-ES as an optimization problem

As discussed in Section 3.3, the query learning task can be viewed as a large optimization problem, where the search space consists of all possible queries that can be presented to the IRS. We convert the query learning task into the problem of finding an optimal Wiki-ES rule which maximizes F-score with respect to the given collection of training documents.

Definition 4.4.1 *(Wiki-ES learning problem).* Let $D_t \subset D$ be the set of training documents for which user has given relevance statements, and let $\bar{Q}$ denote the space of Wiki-ES rules. The learning problem is given by

$$\bar{q}^{\star} = \arg\max_{\bar{q}\in\bar{Q}} F(\bar{q}, D_t)$$

where $F : (\bar{q}, D_t) \mapsto c \in [0, 1]$ is the Wiki-ES fitness function, which corresponds to the F-score within the training document set D_t.

The rationale for defining the learning problem in terms of Wiki-ES rules instead of Wiki-queries stems from the following reasons. The first one is the multi-modality of the user's relevance function. As pointed out by Tamine et al. [63], the relevant documents corresponding to the same topic can be dispersed into different regions of the document space, and thereby have quite different profiles. This implies that in order to recover the relevant documents it is necessary to explore the document space in a number of directions at the same time. Therefore, given the definition of a Wiki-ES rule as a voting system, it appears to be a natural solution for the multi-modality problem as it utilizes a number of Wiki-queries while making the retrieval decisions.

The use of Wiki-ES is also motivated by the fact that unlike classical methods, GP-based approaches always operate with a population of queries rather than a single query. Therefore, we are likely to obtain better results by using several individuals from the population to represent the solution, in case of a multi-modal problem, rather than rely on a single query candidate. Hence in order to solve the above optimization problem, we have chosen to use a co-evolutionary GP approach, where multiple sub-populations are evolved simultaneously to produce Wiki-queries that can be combined to produce an optimal Wiki-ES rule. The details of the algorithm are provided in Section 5.

5. Wiki-ES GP-algorithm

The aim of the proposed GP algorithm is to generate better fit queries using a mechanism inspired by biological evolution [49]. The approach is population based, where each individual represents a Wiki-query. The idea behind the technique is that, for a given population of individuals, the environmental pressure causes natural selection leading to a rise in the fitness of the population. Once the genetic representation of a query and the fitness function is defined, the algorithm proceeds to initialize a population of queries randomly. The population of Wiki-queries is then improved through repetitive application of Selection, Crossover, Mutation and Replacement. To ensure sufficient diversity and reduce the risk of over-fitting the training set, the population is evolved in a number of co-evolving sub-populations. The Wiki-ES rules are then formed by collecting the fittest individuals from each sub-population to form the set of queries that participate in the voting function.

5.1. Genetic representation

Each query is expressed as a syntax tree with the nodes acting as boolean operators and the terminals as the concepts; see Table 1 for correspondence between the common GP components and the Wiki-queries. Fig. 4 shows one such query which acts as an individual in the population. The query shown in the figure is composed of four concepts, $\{w_1, w_2, w_3, w_4\}$, and the basic boolean operators, $\{AND, OR, NOT\}$. The tree represents a boolean expression $(w_1 \wedge w_2) \vee (w_3 \wedge (\neg w_4))$. Such a query will lead to the selection of those documents from the library which either contain the concepts w_1 and w_2 or it contains the concept w_3 but not w_4. Each tree has a depth which is a representative of the size of a tree. The depth of a tree is the number of branches traversed to reach the deepest terminal. The tree in Fig. 4 has w_4 as the deepest terminal and the depth of the tree is 3. It should be noted that the depth of a root node is 0.

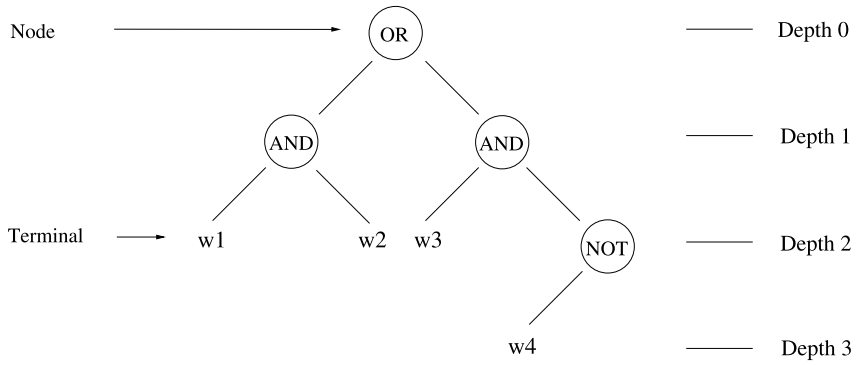

Fig. 4. Genetic representation.

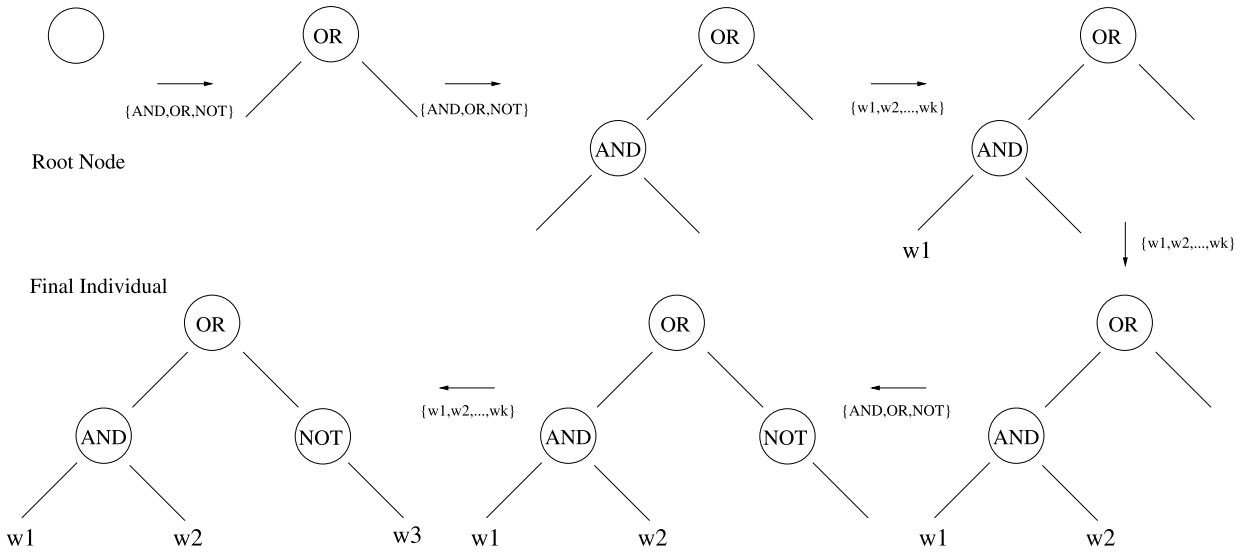

Fig. 5. Random initialization of an individual.

5.2. Population initialization

Like in any evolutionary algorithm, the initial population individuals are generated randomly in genetic programming. The maximum depth (d_{max}), an individual can have, is given as input. A number d is chosen randomly from the set $\{1, 2, 3, \ldots, d_{max}\}$. The chosen number becomes the depth of the tree (individual) to be initialized. Starting from the root node, an operator is chosen randomly from the set $O = \{AND, OR, NOT\}$, and placed at the node. If the node turns out to be *AND* or *OR*, then two subnodes are created; otherwise a single subnode is created. The procedure is repeated for each of the subnodes and the tree size grows. At a depth $d - 1$, a terminal should be chosen to terminate the growth of the tree. Therefore, random choices are made from the set $W_0 = \{w_1, w_2, \ldots, w_k\}$ and the concepts are placed at the terminals. This completes the procedure to generate a single individual. Following a similar procedure, a number of individuals equal to the population size N are generated; the next step is to assign fitness to each individual. Fig. 5 shows the steps involved in initializing an individual of depth 2.

5.3. Fitness assignment

As already mentioned, the set $W_0 = \{w_1, w_2, \ldots, w_k\}$ is created by scanning through the training set of documents and choosing the most relevant concepts which give a good representation of the training set. Once a random query is composed using members from the set W_0 and the basic boolean operators, the query can be evaluated by verifying it against the training set. The boolean query is applied to each of the document in the training set, and the query predicts the document as relevant or irrelevant. The number of correct relevant or irrelevant predictions leads to the fitness for the query. The

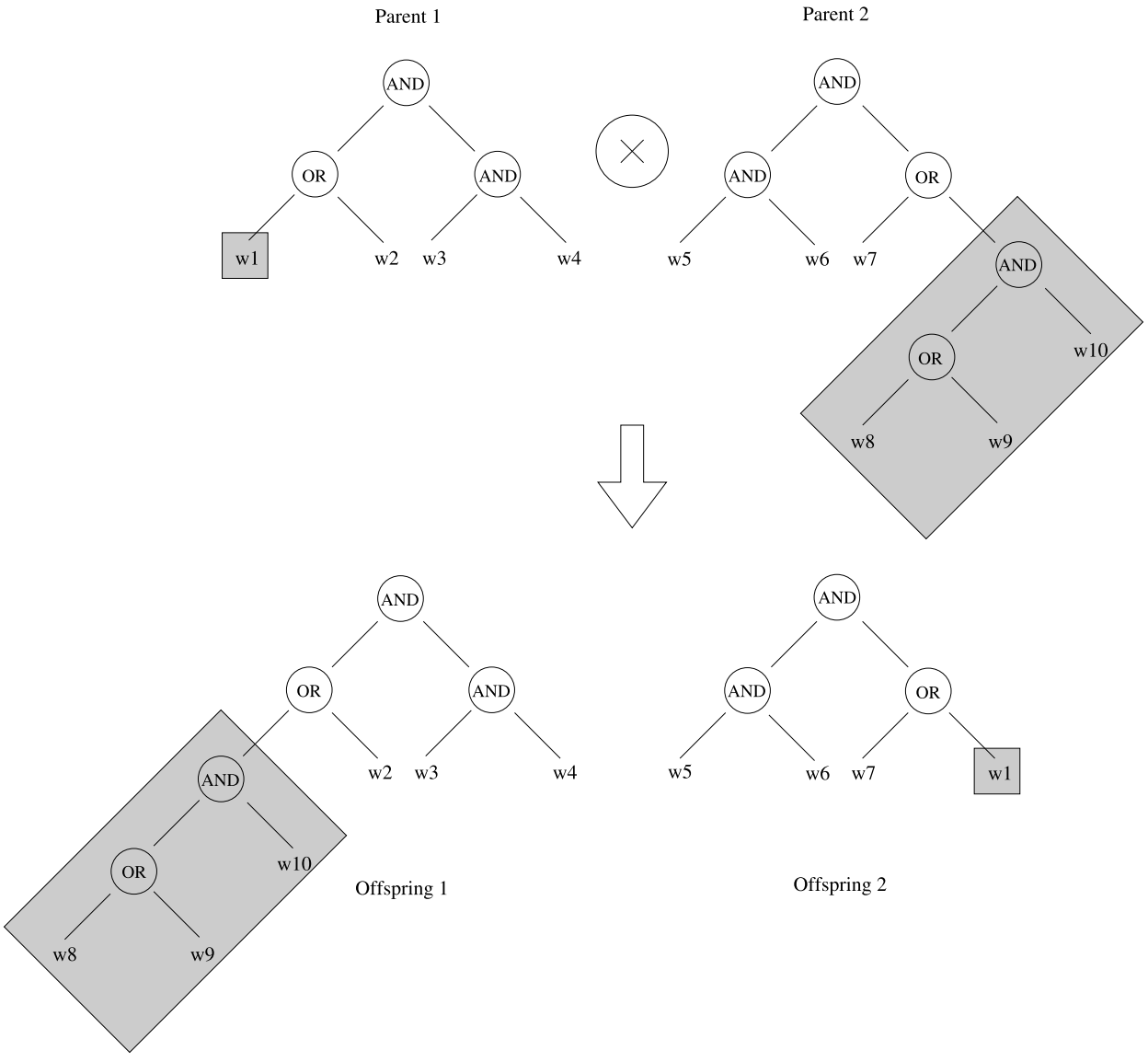

Fig. 6. Crossover.

algorithm searches for those queries which provide the maximum F-score. Degeneracy often exists, as there is a possibility of more than one query producing the same results and therefore having the same fitness.

5.4. Producing new queries

New queries or offsprings are produced from the parent queries by means of crossover and mutation. A crossover method is chosen such that two parents result in two offsprings. The crossover is performed by randomly choosing a crossover point in each parent tree. Once the crossover points are chosen, the offsprings are created by swapping the subtree rooted at the crossover point of one parent with the subtree rooted at the crossover point of the other parent. Fig. 6 shows two parents and the crossover operation. The subtrees to be swapped are shown shaded in the figure. Swapping the two shaded subtrees produce the offsprings.

Once the crossover operation is performed and the offsprings are produced, they undergo a mutation operation. A point mutation operation has been used where each node is considered in turn, and with a particular probability the primitive stored at the node is replaced with another randomly chosen primitive of the same arity.[1] The mutation operation has been

[1] Arity means the number of arguments a function can take. In a query, a *NOT* gate cannot be mutated with an *OR* or *AND* gate as *NOT* takes a single argument as input and on the other hand *AND* and *OR* take two arguments as input.

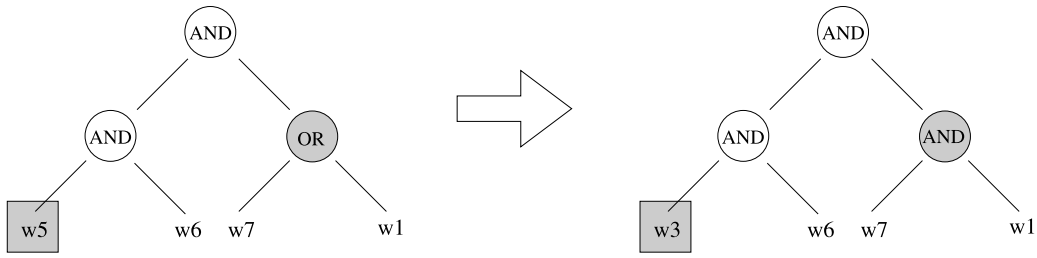

Fig. 7. Mutation.

shown in Fig. 7 for the second offspring produced from crossover. Making a choice based on a mutation probability, the nodes with w_1 and *OR* get chosen. w_1 is replaced by a random member from the set, $\{w_1, w_2, \ldots, w_{10}\}$ and *OR* is replaced by a random member from the set $\{OR, AND\}$. The crossover and mutation operation together produce the offsprings which compete with other members to enter the population based on their fitness.

5.5. Algorithm description

The proposed algorithm follows the framework of a general evolutionary algorithm. Instead of having a single population, the algorithm maintains multiple sub-populations which interact with each other during the optimization run. The algorithm terminates when the prescribed number of generations are completed. At the end of the optimization run, the algorithm provides elites from each of the sub-populations as final solutions. These elites are expected to represent different niches in the search space. Each elite represents a Wiki-query which participates in the formation of a Wiki-ES rule. Multiple queries are accepted as solutions from the algorithm, as we do not wish to rely on a single query. For any document, output of each query is taken into account through the voting function and the decision for relevance or irrelevance is made. A flowchart for the proposed genetic programming algorithm has been presented in Fig. 8. In the following, we also discuss a stepwise procedure for implementing the algorithm.

1. Initialize M different sub-populations randomly. Each sub-population contains n number of individuals. It is noteworthy that the choice of M determines the number of Wiki-queries participating in the Wiki-ESR rule, i.e. $M = |A|$ in Definition 4.3.7.
2. Assign fitness to all the initialized individuals.
3. Initialize a generation counter $Gen = 0$.
4. If Gen is less than maximum number of prescribed generations then go to Step 5, otherwise go to Step 16.
5. Increment the generation counter by 1, $Gen = Gen + 1$.
6. Initialize a sub-population counter $S = 0$.
7. If S is less than number of sub-populations M then go to Step 8, otherwise go to Step 4.
8. Increment the sub-population counter by 1, $S = S + 1$.
9. Initialize an offspring counter $Off = 0$.
10. Choose two individuals randomly from sub-population S, perform a tournament and choose the better individual as one of the members for crossover.
11. Generate a random number between 0 and 1. If the value is less than $1/M$, then choose two individuals randomly from sub-population other than S, otherwise choose two individuals randomly from the sub-population S. Perform a tournament and choose the winner as the other member for crossover.
12. Perform crossover with a crossover probability p_c. This produces two offsprings.
13. Mutate the offsprings with a mutation probability p_m.
14. Increment the offspring counter by 2, $Off = Off + 2$.
15. If offspring count, Off is equal to n, then combine the offsprings and the individuals from the sub-population S into a pool. Choose the n best members from the pool, copy it into the sub-population S and go to Step 7. If offspring count, Off is less than n then go to Step 10.
16. Choose the best members from each sub-population as final solutions.

5.6. Formation of Wiki-ES rules

As already mentioned, the suggested GP algorithm produces multiple queries as its output. If the number of sub-populations is M, then the number of final queries is also M. Given a document, each query suggests it as either relevant or irrelevant. However, we wish to take a weighted contribution of each of the queries before making a final decision. Let each of the query be represented by q_i: $i \in \{1, 2, \ldots, M\}$ and the associated fitness be represented by F_i: $i \in \{1, 2, \ldots, M\}$. For any given document d, if we need to decide whether it is relevant or irrelevant, output of each of the query is considered. Let the output of each query for the document d be b_i: $i \in \{1, 2, \ldots, M\}$, where b_i is either 0 or 1. Now a weighted contribution of the queries is accounted in the following metric μ:

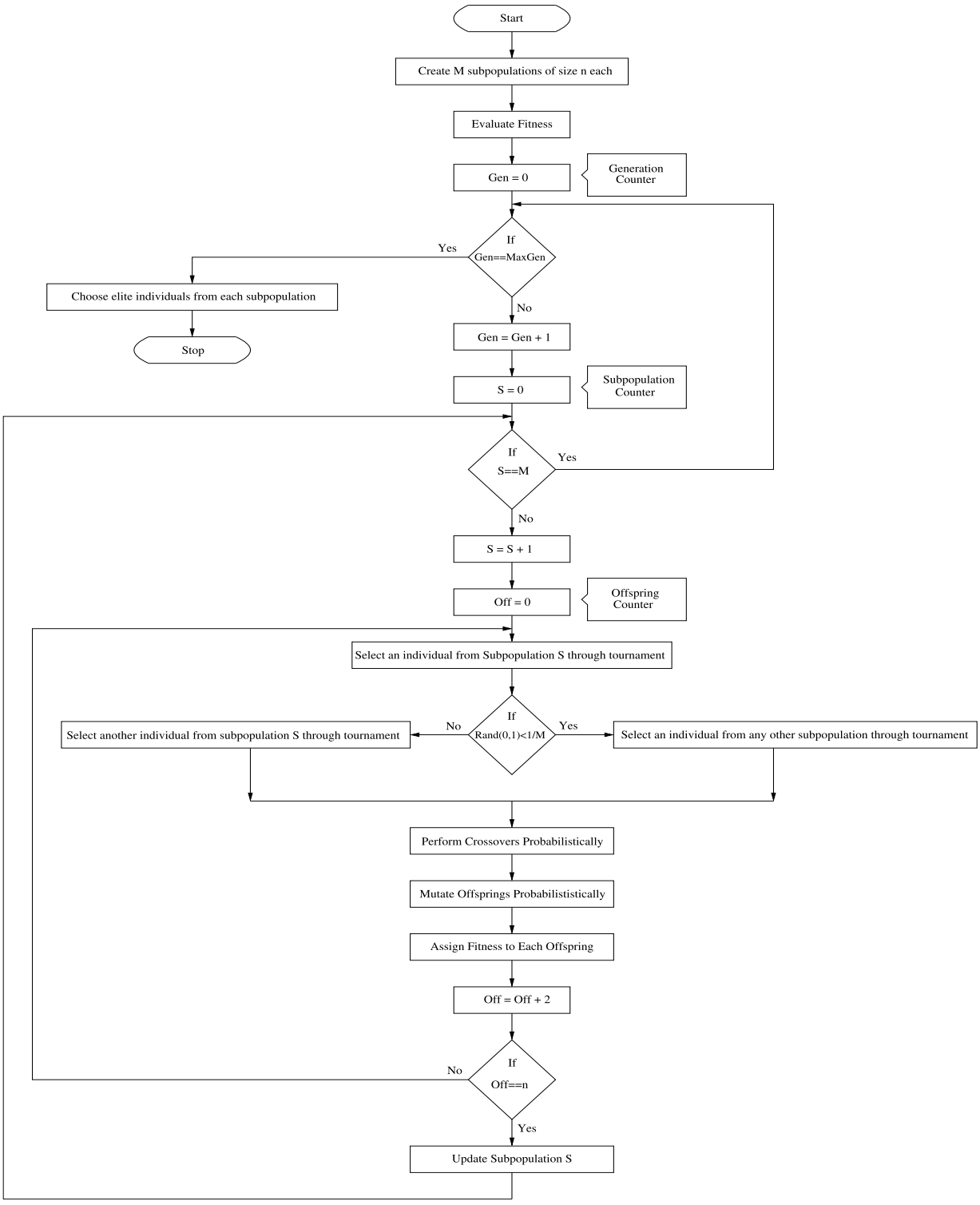

Fig. 8. Flowchart for GP algorithm.

$$\mu = \frac{\sum_{i=1}^{M} F_i b_i}{\sum_{i=1}^{M} F_i}. \quad (1)$$

If the value of the metric μ is greater than 0.5 then the document is considered relevant, otherwise it is considered irrelevant. Using this weighted contribution, the information from various niches are taken into account and overfitting of a query to the training document set is also avoided.

6. Experiments and results

The purpose of QBE frameworks such as Wiki-ES is to generate query expressions that are optimal in terms of precision and recall. The way how the query expressions are then used can take different forms ranging from document filtering tasks on a single system to searching on multiple third party engines. Given that the quality of the query expressions themselves is difficult to evaluate as such, we examine the performance of the proposed system when applied to a simple document filtering task using the topics in TREC-11 corpus. Sometimes the notion of filtering is defined in very broad terms. For instance, Belkin and Croft [1] interpret filtering as "a variety of processes involving the delivery of information to people who need it". In this paper, we consider filtering as a continuous document classification task where an influx of documents is to be labeled as relevant or irrelevant according to their relatedness to the user's topic of interest. The filtering experiment is motivated by the natural interpretation of Wiki-ES query expressions as binary classification rules which can be used to categorize the incoming documents as relevant or irrelevant. An additional benefit of using a document filtering task for evaluation is the availability of different baseline models that allow comparison with other non-QBE filtering frameworks.

The experiment is structured as follows. First, we begin with description of the data set in Section 6.1, which is followed in Section 6.2 by an account on the software components used to implement the Wiki-ES system and the parameter setup of the GP algorithm. A description of the other frameworks used in the experiments along with a short discussion on their relevance is outlined in Section 6.3. The results from the comparison of Wiki-ES against competing algorithms are presented in Section 6.4. In particular, we illustrate the benefits of using Wikipedia-concepts for query learning by benchmarking the performance of Wiki-ES against a corresponding term-based model.

6.1. Data

The documents included in TREC-11 corpus are Reuters RCV1 news stories from years 1996-1997. The data is partitioned into a training set (items dated between 1996-08-20 to 1996-09-30) and a test set (remainder of the collection). The training and test set are further divided into 100 topic-specific subsets. All 100 TREC-11 topics (numbered R101–R200) are used in the experiment. In this paper, only the initial training data is used, while the relevance statements available for adaptive learning are not utilized. Also none of the information in the separately available topic description file is used.

Given that query learning techniques tend to be highly dependent on the quality and amount of training data, it is worthwhile to take a closer look at the data available for the 100 TREC-11 topics. Fig. 9 shows two histograms displaying the number of training and evaluation documents for each topic. To describe how data sets are balanced between relevant and irrelevant documents, the frequency bars are split to reflect their proportions in both data sets. On average there are 12 relevant and 39 irrelevant document examples in the training data, and 90 relevant and 713 irrelevant in the evaluation set. However, the variation between topics is quite drastic, especially in the evaluation set. As it can be seen from the histogram, the first 50 topics have a large evaluation set as compared to the remaining topics. It can also be seen that some topics are highly imbalanced, in the sense that there is only a handful of relevant documents for hundreds of irrelevant items, e.g. in the case of topic R137 less than 1% of the documents are relevant in the evaluation set. Then on the other extreme, a few topics (e.g. R175) are very loosely defined with majority of the documents being relevant. When considering the performance of the Wiki-ES model, as well as the benchmarks, both the quantity and balance of training data play important roles. In general, topics with relatively large proportion of relevant examples in the training data fare better than the ones with very few relevant items. The topics with few relevant documents provide good test-cases for evaluating the efficacy of the algorithm.

6.2. Wiki-ES implementation and parameters

The system used in the experiment was implemented using Java-based software on top of the GATE platform, which provides tools for standard document preprocessing tasks. The other software components used in the implementation and evaluation of Wiki-ES framework are described as follows:

- Wikipedia-model: The Wikipedia-based content model was built using the WikipediaMiner published by Milne et al. [41], which was suitably modified and integrated into our framework.
- NER: The named-entity recognition task was carried out using a Conditional Random Field (CRF) classifier proposed by Finkel et al. [15], which is trained on CoNLL 2003 dataset.
- Genetic programming: The co-evolutionary GP algorithm described in Section 5 was implemented using the JGAP toolbox provided by Meffert et al. [36].

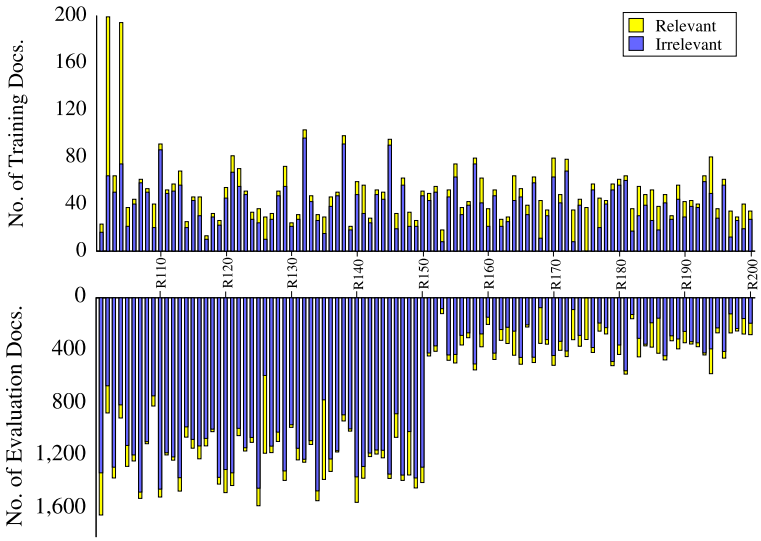

Fig. 9. The number of relevant/irrelevant documents in TREC-11 topics.

Table 2
GP parameters.

Parameter name	Value
Number of generations, G	250
Number of sub-populations, M	10
Sub-population size, N	100
Crossover probability, p_c	0.9
Mutation probability, p_m	0.9
Initial tree depth	4
Maximum crossover depth	8

The GP procedure used in the paper has the usual genetic programming parameters like population size, crossover probability, mutation probability, etc. The parameter setting used in this experiment is given in Table 2.

In addition to the general GP parameters, we have used 15 as the maximum size for the terminal set while constructing query trees. That is, when building the queries, the maximum number of different Wikipedia-concepts that could appear in a single Wiki-query was limited to 15. The choice of Wikipedia-concepts for each topic was carried out by selecting the ones that appear most frequently in the relevant training documents.

6.3. Benchmark frameworks used in experiments

The document filtering task presented by TREC-11 can be solved in a variety of ways. Therefore, for evaluating the proposed system, a number of benchmarks need to be considered. To provide a brief description of the alternative frameworks, we will begin the discussion with QBE paradigm, and then have an overview of the other methods thereafter.

6.3.1. IQBE benchmarks

In the recently introduced QBE systems (Cordón et al. [13], López-Herrera et al. [30,29]), the query generation task is performed by using genetic programming techniques, where the objective of a GP algorithm is to maximize a performance metric like recall, precision, F-score or other variants. When two conflicting performance measures like precision and recall are optimized at the same time, a multi-objective GP algorithm is required. Therefore, the QBE algorithms can differ based on the objective(s) being optimized. The GP algorithms can also differ from each other in terms of the feature space they use to construct the queries. For example, there can be a GP-algorithm which optimizes a performance measure to come up with a token based query, and on the other hand, there can be a GP-algorithm which optimizes a performance measure to come up with a concept-based query. To summarize, the QBE algorithms can be categorized according to the following criteria: (i) the objective(s) to be optimized; or (ii) the type of feature space used.

The recent QBE systems, such as Cordón et al. [13] and López-Herrera et al. [30,29], have all used multiple objectives (recall, precision) to generate the queries. The difference between the two paradigms is that the single-objective methods yield only a single optimal query, whereas the multi-objective methods produce several queries representing optimal trade-off solutions (Pareto-frontier). It is noteworthy that the query produced by the single-objective method corresponds to a particular point on the Pareto-frontier generated by a multi-objective method. In the context of this paper, we desire to select

the query that maximizes F-score. If multi-objective-method is used for this purpose, it would mean generating the entire Pareto-frontier and then choosing the solution with highest F-score. The same solution can be obtained by implementing a single objective method that maximizes the F-score. Hence, the utility of the multi-objective algorithm lies in the context where a decision has to be made.

In this paper, we want to compare the state-of-the-art algorithms, such as López-Herrera et al. [30], against Wiki-ES in a single-objective context. For this purpose, we have implemented Token-GP algorithm which uses the same feature space as the state-of-the-art algorithms but optimizes a single objective, i.e. F-score. In effect, Token-GP produces the Pareto-optimal solution corresponding to the maximum F-score. The algorithm relies on a similar genetic programming framework as Wiki-ES. The parameter values are described in Table 2. Hence, the algorithm is essentially the same as Wiki-ES with the following modifications: (i) use of lemmatization and stop-word removal in the preprocessing stage, which is a common practice in the QBE systems; (ii) replacing d-rel with a simple binary-valued function which gives 1 if a token is found in the document and 0 otherwise.

The Token-GP algorithm is effectively a single objective version of the recent bag-of-words based QBE frameworks, where the only difference being the number of populations used in learning. The standard QBE frameworks are single population algorithms, whereas Token-GP uses co-evolutionary learning. The benefit of using this technique is mainly in reducing the over-fitting tendency of the outcome as the result is stabilized across multiple learnt queries. Hence, the obtained optimal result for the Token-GP algorithm is directly comparable to the results that are expected to be obtained by using contemporary single-population GP-frameworks.

6.3.2. Alternative benchmarks

In addition to the QBE frameworks discussed in this paper, there are a number of alternative approaches to document filtering which do not rely on boolean query expressions. Below is a brief summary of the methods that have been included as benchmarks. The approaches are categorized by the type of the algorithm used.

(i) *Kernel methods*: Techniques based on support vector machine (SVM) are commonly found to be top performers in classification tasks. For comparison, three variants of SVM have been included. As naive baselines, we consider linear SVMs; one trained on bag-of-words profiles (Token-SVM) and another one trained on Wikipedia-concept profiles (Wiki-SVM). To provide a more sophisticated benchmark, we include the KerMIT algorithm of Cancedda et al. [6] which is based on combination of SVMs and perceptrons. The measure it optimizes is F-score.
(ii) *Decision trees*: Another commonly applied classification approach is the decision tree algorithm C4.5. In this paper, two such classifiers are included: a bag-of-words based C4.5. (Token-C4.5) and its concept-based variant (Wiki-C4.5).
(iii) *Multicriteria information filtering*: Recently, Bordogna and Pasi [2] have proposed a flexible multicriteria information filtering model (flexible-PENG) that allows customization according to personal interests and context of users. The system has been successfully applied to document filtering, and the results have been included in the comparisons.
(iv) *Clustering methods*: There are also a few clustering methods that can be effectively applied to document filtering. One of them is $\alpha\beta$-Neighborhood method of Fonseca-Bruzón et al. [17], which uses a modified Nearest Neighbor classifier to solve a binary classification task. The approach is based on the idea that documents used in learning can form internal subdivisions which needs to be taken into account while classifying new documents.
(v) *Profile adaptation*: A number of commonly used document filtering methods have been inspired by Rocchio-like query expansion. As examples of such approaches, we have included the CAS-ICT framework by Xu et al. [65], CMU framework by Collins-Thompson et al. [11], and the incremental profile learning approach (IRIT-SIG) of Boughanem et al. [5].

6.4. Results

In this section, we present the results from three experiments carried out using TREC-11 data. The first experiment, discussed in Section 6.4.1, examines the importance of using Wikipedia-concepts in Wiki-ES rules by comparing them against the results obtained by running the same algorithm with bag-of-words document model. By using the bag-of-words profile in the competing model we get an effective comparison against the established IQBE-paradigm. The second experiment, presented in Section 6.4.2, evaluates the Wiki-ES model against the well-known classification models, Support Vector Machines (SVM) and the decision-tree algorithm C4.5. The third experiment compares the performance of the Wiki-ES algorithm with contemporary document retrieval frameworks.

6.4.1. Experiment 1: Effect of Wikipedia semantics

Given that the main contribution of the Wiki-ES framework is the integration of Wikipedia's knowledge into the query learning problem, the first question to ask is: how much the retrieval results have been improved by the infusion of the semantic information. In order to quantify the effect, we consider an experiment where the co-evolutionary GP-algorithm is run with two alternative content models: the Wikipedia-based model and the bag-of-words model. This allows us to eliminate the effect of the algorithm and focus on the improvement following from the concept-based representation of documents and queries.

The key performance measures are summarized in Table 3, where Token-GP refers to the model using the bag-of-words representation. The results are computed as averages across all 100 topics. A direct comparison shows that Wiki-ES yields an

Table 3
Results for Wiki-ES and Token-GP algorithms.

Algorithm	F-score	Precision	Recall	Accuracy
Wiki-ES	0.4218	0.4104	0.5200	0.8436
Token-GP	0.2596	0.4002	0.2925	0.8466

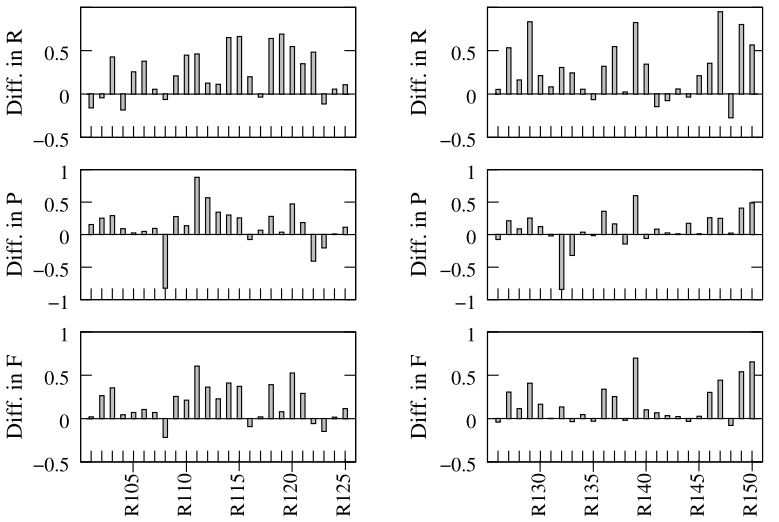

Fig. 10. Differences in F-score (F), Precision (P), and Recall (R) between models Wiki-ES and Token-GP for topics R101–R150.

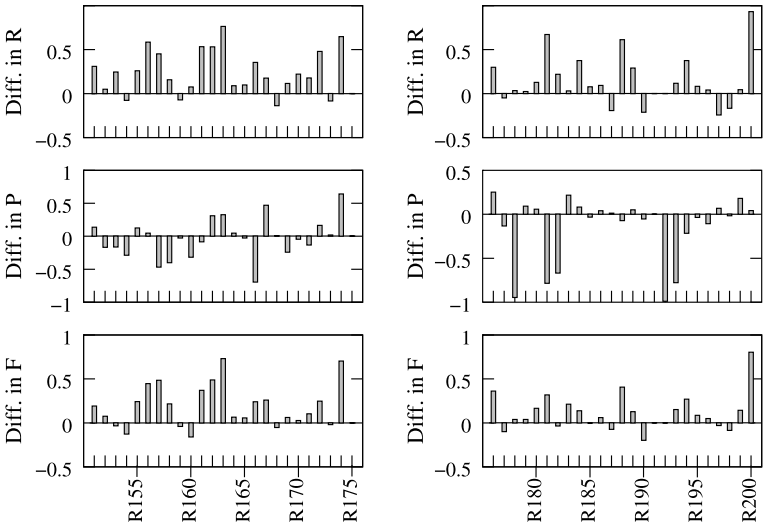

Fig. 11. Difference in F-score (F), Precision (P), and Recall (R) between models Wiki-ES and Token-GP for topics R151–R200.

improvement of 62% in F-score when compared with the Token-GP model. Interestingly, when comparing the results with respect to precision and recall, we find that most of the reported difference in F-score is due to better recall of Wiki-ES, while precision and accuracy are roughly the same. After all, recognizing the way how the concept-relatedness measure is utilized in the evaluation of Wiki-queries, the outcome was anticipated due to the ability of Wiki-queries to match such documents as well which contain a closely related concept that would have been ignored by a word based search. On the other hand, the Token-GP based rules require that words in the query expressions are directly detected, which is likely to weaken their ability to match relevant documents.

To provide a better idea of the F-scores obtained from the two algorithms across individual topics, Figs. 10 and 11 show the difference in F-scores for Wiki-ES and Token-GP. Positive bars in the figures indicate the topics where the use of Wikipedia's semantics has been beneficial in terms of F-score, recall and precision. The reason for splitting the evaluation into subfigures stems from the characteristics of the topics. The first half of the dataset (R101–R150; Fig. 10) represents topics where the individual query expressions participating in the Wiki-ES rules tend to have more complicated structures.

Table 4
Results for Wiki-ES, Token-C4.5, Token-SVM, Wiki-C4.5 and Wiki-SVM.

Algorithm	F-score	Precision	Recall	Accuracy
Wiki-ES	0.4218	0.4104	0.5200	0.8436
Token-C4.5[a]	0.2849	0.2770	0.3730	0.8048
Token-SVM[b]	0.2215	0.5755	0.2098	0.8863
Wiki-C4.5[a]	0.3150	0.3478	0.3678	0.8386
Wiki-SVM[b]	0.2530	0.5649	0.2290	0.8868

[a] Default WEKA [22] implementation for C4.5 algorithm, where information gain is utilized to choose the attributes.
[b] Uses the LIBSVM [8] implementation for maximum margin classification. No threshold optimization is considered.

Table 5
Performance matrix showing the performance of each algorithm when compared with the other algorithms. The comparison is computed as the relative difference in F-scores, $100 \times (F_{algo1} - F_{algo2})/F_{algo2}$, where F_{algo1} is the average F-score of the algorithm in the column and F_{algo2} is the average F-score of the algorithm in the row.

Algorithm	Wiki-ES	Token-GP	Token-C4.5	Token-SVM	Wiki-C4.5	Wiki-SVM
Wiki-ES	0%	–	–	–	–	–
Token-GP	62.48%	0.00%	–	–	–	–
Token-C4.5	48.07%	−8.87%	0.00%	–	–	–
Token-SVM	90.43%	17.21%	28.61%	0.00%	–	–
Wiki-C4.5	33.91%	−17.58%	−9.56%	−29.68%	0.00%	−19.67%
Wiki-SVM	66.69%	2.60%	12.58%	−12.46%	24.48%	0.00%

In particular, they commonly feature conditions that would require the use of NOT-gate to construct the query expressions. For example, in topic R120, we are looking for documents on deaths of mine workers where the death has occurred due to a mining accident and is not related to an ethnic clash between miners. When comparing the performance differences, it appears that the Wikipedia-based approach has the largest edge over Token-GP within the first 50 topics. For topics R101–R125 the average percentage improvement in F-score is 91.37% and 82.51% for topics R126–R150 in favor of Wiki-ES, which are both considerably larger than the improvement across all of the topics. The results reported for the remaining topics (R151–R200) show that the use of Wikipedia-concepts has improved the F-scores substantially for these topics as well; see Fig. 11. However, the average percentage difference in F-score is 54.57% for topics R151–R175 and 38.57% for R176–R200.

To summarize, the experiment lends support to the conclusion that the use of Wikipedia's concept information appears to have a substantial effect on the performance of the Wiki-ES framework. The improvement stems from the ability of the rules to achieve higher recalls without losing too much precision.

6.4.2. Experiment 2: Comparison with standard SVM and C4.5

The purpose of the second experiment is to compare the performance of Wiki-ES model against two well-known classification algorithms, SVM and C4.5. In order to evaluate the effect of feature selection as well, the benchmark algorithms are trained using both token-based (bag-of-words) document representations and Wiki-based document model. The support vector algorithms are referred to as Token-SVM and Wiki-SVM, and the decision-tree algorithms are denoted by Token-C4.5 and Wiki-C4.5, respectively. The classifiers considered here are implemented as standard SVM and C4.5 methods without optimizing them for any specific criteria. The algorithms with optimal threshold selection mechanisms are discussed in the context of the third experiment in Section 6.4.3.

The results are summarized in Table 4 where key performance measures are reported for each of the 5 models. A general comparison of the models suggests that the Wiki-ES framework consistently outperforms its benchmarks in terms of F-score. Once again, the primary cause for the performance advantage appears to be the improved recall of Wiki-ES rules. Whereas SVM-based models appear to yield better results if only precision would be considered. However, the recalls of Token-SVM and Wiki-SVM are quite poor, which leads to an overall modest performance. The differences in accuracies are relatively small for all of the models.

Finally, to consider the effect of training data on the benchmark algorithms, we have computed relative differences in F-scores between each pair of models. The results are presented in Table 5. For the sake of completeness Token-GP is also included in the comparison. A quick overview suggests the following observations. First of all, we find that the use of Wikipedia-concepts in document models had a positive effect on the results for all the algorithms. Moreover, there is a substantial difference in the size of the effects. The effect of Wikipedia-concepts is large between Wiki-ES and Token-GP, but the corresponding comparisons for pairs Token-SVM vs Wiki-SVM and Token-C4.5 vs Wiki-C4.5 show only modest improvements. This is best explained by the fact that SVM and C4.5 based algorithms are not able to use concept-relatedness information efficiently while classifying documents into relevant or irrelevant. Overall, the results indicate that concept-based information is useful under these evaluation settings which motivates further development in this direction.

6.4.3. Experiment 3: Comparison with general document filtering models

As discussed in Section 6.3.2, a number of alternative document filtering paradigms have been proposed, which do not rely on boolean query learning. In this experiment, the two QBE algorithms (Wiki-ES and Token-GP) are compared against

10 alternative approaches which range from optimized kernel methods to query expansion and clustering. Given that most of the approaches have been originally designed with TREC-11 criteria in mind, the comparison is done with respect to the two measures proposed in TREC-11 conference; see the final report by Robertson and Soboroff [56]. The first one is F_β-measure defined by van Rijsbergen [55], which is a function of recall and precision with a free parameter β to determine the relative weighting of recall and precision:

$$F_\beta = \left(1+\beta^2\right)\frac{PR}{(\beta^2 P)+R}$$

where R and P denote recall and precision, respectively. By selecting the value of $\beta = 0.5$, we obtain the T11F-measure used by TREC. The second TREC-measure is the linear utility T11U[2]:

$$\text{T11U} = 2 \times \text{No. of relevant docs retrieved} - \text{No. of irrelevant docs retrieved}$$

which corresponds to a simple retrieval rule that is equivalent to filtering the documents with estimated probability of relevance greater than 0.33.

The results of the experiment are given in Table 6. The table is divided into two parts according to the number of topics considered. The first part, Panel A, gives the aggregated performance figures across all 100 TREC-topics. The second part, Panel B, shows a decomposition of the result into two topic groups: the assessor topics (R101–R150) and the intersection topics (R151–R200). As discussed by Robertson and Soboroff [56], the separation is motivated by the use of different techniques to construct the topics. The first 50 topics are defined by the assessors at NIST, whereas the remaining 50 were built as intersections of document category assignments specified by the journalists at Reuters.[3] The results reported for the TREC-11 baselines have been collected from batch and adaptive task documents published on the TREC filtering track website [64]. The TREC-11 run identifiers of the baseline results are provided in the last column of Panel B. The figures for the other two competitors, $\alpha\beta$-Neighborhood method and Flexible-PENG, are obtained from the articles by Fonseca-Bruzón et al. [17] and Bordogna and Pasi [2], respectively.

All of the models included in the comparison have been optimized for either of the two measures T11F or T11U. In addition to the performance measures, the table indicates the category of each model, the training set, and the number of topics with zero returns. The models and their categories are described in Section 6.3.2. The training sets (batch, adaptive) represent two document filtering subtasks of TREC-11. Both batch and adaptive tasks include a small initial collection of relevance statements that are used for training the filtering model, however, the adaptive task also includes an additional dataset that can be used for reinforcement learning. The incorporation of results from adaptive task as well is motivated by the TREC-11 results, which suggest that reinforcement learning methods tend to outperform batch systems after going through an adaptation phase. The models included in the comparison are among the top-performers of both tasks.

A comparison of the aggregated performance figures in Panel A shows that Wiki-ES is a tough competitor in terms of both reported criteria. When only T11F is considered, Wiki-ES achieves the highest score. In terms of T11U, it has the second highest score and is outperformed only by the recent $\alpha\beta$-Neighborhood based adaptive filtering method by Fonseca-Bruzón et al. [17]. However, it should be noted that Wiki-ES has not been optimized for T11U but T11F. Another difference is the type of training data used; Wiki-ES is a batch algorithm whereas $\alpha\beta$-Neighborhood is an adaptive algorithm. In general, when T11U measure is considered, the adaptive filtering algorithms appear to perform better than the models trained on batch data only. It is also worthwhile to recall that although all of these algorithms can be applied for document filtering, they have been designed with different platforms and applications in mind. For example, in the QBE frameworks the goal is not only to achieve high filtering performance but also to learn query expressions which can be transferred to other platforms that enable boolean search. Therefore, given that Wiki-ES is a batch algorithm and works on a QBE framework, we find our preliminary results encouraging.

In order to get further intuition on the source of performance differences, it is worthwhile to consider results decomposed into assessor and intersection topics shown in Panel B. The results enable a direct comparison of the algorithms in the spirit of the final TREC-11 report by Robertson and Soboroff [56]. A quick glance at the panel is sufficient to make a few interesting observations. First, when considering the average figures across both topic groups, we find that our approach performs quite steadily regardless of the topic group. The same conclusion holds for the top-performing competitor, the $\alpha\beta$-Neighborhood technique of Fonseca-Bruzón et al. [17]. In terms of T11U measure the Flexible-PENG method by Bordogna and Pasi [2] is also a stable performer. However, when examining the behavior of the remaining baselines, the variation between topic groups appears to be considerably larger. Though they have shown outstanding performance on the assessor topics, there is a clear drop in the results reported for the intersection topics. This obvious difference is also verified in the report by Robertson and Soboroff [56], where box-plots of the results have been reported separately for the two topic groups. The performance gap is large enough to tilt the aggregated figures of Panel A in favour of the algorithms with steadier overall performance.

[2] In the official definition the linear utility T11U is replaced by a scaled utility T11SU = (max(T11U/MaxU, MinNU) − MinNU)/(1 − MinNU), where MaxU = 2 ∗ (Total number of relevant documents) and MinNU = −0.5.

[3] According to Robertson and Soboroff [56], the Reuters' rules for category assignment specify that at least one category must be assigned to each document. Additional categories are used when considered immediately relevant or in case of uncertainty about the correct category.

Table 6
Comparison with general document filtering models. The results for top three algorithms are bolded.

Panel A: Performance comparison for all 100 topics (R101–R200)

Model	T11F	T11U	Zeros	Optimized	Training set	Category
Wiki-ES	**0.391**	**0.406**	3	T11F	Batch	QBE-GP
Token-GP	**0.317**	0.275	21	T11F	Batch	QBE-GP
$\alpha\beta$-Neighborhood	–	**0.477**	–	T11U	Adaptive	Clustering
Flexible-PENG	**0.302**	0.402	0	T11U	Adaptive	Profile adapt.
KerMIT (batch)	0.298	0.375	5	T11U	Batch	Kernel
KerMIT	0.237	0.372	0	T11F	Adaptive	Kernel
IRIT/SIG	0.273	0.361	0	T11U	Batch	Profile adapt.
ICT (batch)	0.090	0.340	66	T11U	Batch	Profile adapt.
ICT	0.245	**0.403**	4	T11U	Adaptive	Profile adapt.
CMU	0.220	0.362	0	T11F	Adaptive	Profile adapt.
CMU	0.222	0.369	0	T11U	Adaptive	Profile adapt.

Panel B: Performance comparison for assessor topics (R101–R150) and intersection topics (R151–R200)

	Assessor topics			Intersection topics			TREC run-id
Model	T11F	T11U	Zeros	T11F	T11U	Zeros	
Wiki-ES	0.319	0.351	1	**0.463**	**0.461**	2	–
Token-GP	0.199	0.203	11	**0.434**	0.347	10	–
$\alpha\beta$-Neighborhood	–	**0.464**	–	–	**0.490**	–	–
Flexible-PENG	0.424	0.409	0	**0.179**	**0.395**	0	–
KerMIT (batch)	**0.495**	**0.505**	2	0.101	0.245	3	KerMITT11bf2
KerMIT	0.426	0.458	0	0.048	0.285	0	KerMITT11af3
IRIT/SIG	**0.455**	**0.485**	0	0.091	0.237	0	iritsigb
ICT (batch)	0.180	0.350	16	0.000	0.330	50	ICTBatFT11Ua
ICT	**0.428**	0.475	0	0.062	0.330	4	ICTAdaFT11Ub
CMU	0.401	0.431	0	0.038	0.293	0	CMUDIRFDESC
CMU	0.410	0.447	0	0.034	0.290	0	CMUDIRUml

Many potential explanations for the performance differences could be conjectured. One version has been offered by Fonseca-Bruzón et al. [17], who consider that the intersection topics are perhaps less homogeneous than the assessor topics, and thereby represent a real-life situation where the user's information need is satisfied by documents coming from different sources. A somewhat similar hypothesis has bee suggested by Robertson and Soboroff [56], who speculate that the Reuters' category labeling system may have been inconsistent which makes the intersection topics a bit fuzzier compared to the assessor judgements. In addition to the labeling noise, another explanation might be the high variation in the generality of the topics which could make learning of larger and more diverse topics harder for systems that expect consistent statements and high degree of homogeneity. Nevertheless, given that in reality it is quite hard to control the quality of topic definitions supplied by the users, it is important not to sacrifice the overall robustness of a system in the favor of highly optimized performance for certain topic types. As pointed out by Fonseca-Bruzón et al. [17], less homogeneous and diverse topics are likely to be encountered in real environments, and therefore systems should be prepared to deal with them correctly.

In summary, a comparison of Wiki-ES against other frameworks indicates that with the help of concept-based information QBE paradigm is competitive even when compared to modern profile adaptation techniques. We also find that the performance of the system has been quite stable regardless of the topic types considered. Hence, although the QBE frameworks such as our Wiki-ES have been mainly designed as expression-learning techniques, the paradigm provides a flexible foundation for solving different types of retrieval tasks. For instance, these preliminary comparisons suggest that QBE frameworks can be effectively employed as robust filtering tools when appropriately modified to account for semantic information and the considerable noise commonly involved in the expression learning process.

7. Conclusions

The purpose of any automated query learning system is to help the user define a query that finds the items relevant to her topic. The conventional frameworks have approached the problem by using a variety of techniques based on literal term matching. However, they have been largely criticized for their inability to account for semantic similarities that are obvious to human readers. One of the problems is the variability of word usage. The same word can mean different things depending on the context. Another well-known problem is the abundant use of synonyms in natural language. Even within small expert domains, there are numerous ways to express the same meaning. To alleviate such vocabulary problems, there has been increasing interest to explore concept-based information retrieval techniques. As far as we know, the present work represents a pioneering step towards utilizing semantic information using Wikipedia within automated query expression learning. In this paper, we propose a new co-evolutionary genetic programming framework, where Wikipedia is used to provide the system with human-and-society level information. Given that Wikipedia is a free and universally available

database of information, the approach has low implementation costs. The active Wikipedia-community also ensures that the knowledge remains updated, which addresses the maintenance concerns commonly encountered with knowledge-based retrieval techniques.

In order to evaluate the performance of the proposed Wiki-ES framework, a number of experiments were carried out to examine both the relevance of semantic information as well as to provide a comparison with a variety of document filtering paradigms. In light of the given evaluation settings, we find that the use of concept-based information helps to improve the retrieval performance. The effects get strongly pronounced especially when comparing the Wikipedia-based approaches against their bag-of-words counterparts that lack semantic information about synonyms and related concepts. In particular, the use of concepts-based approaches can be beneficial when trying to enhance the recall of boolean query expressions. To further explore the method's performance, comparisons are drawn against different forms of profile adaptation and reinforcement learning methods designed for document filtering. Although the QBE framework is more an expression learning method than a pure document filtering technique, the Wiki-ES framework turned out to be a tough competitor in terms of aggregated recall and precision measures. Overall, the results of these preliminary experiments have been promising, which motivates further exploration on the use of concept-based information to improve retrieval performance. In the future work, we plan to investigate how concept relatedness information can be incorporated into paradigms other than QBE.

Acknowledgements

The authors would like to thank Emil Aaltonen Foundation and Finnish Cultural Foundation for their support. Part of the work was supported by Academy of Finland research grant no. 133387. The authors would also like to express their thanks to the editors and reviewers for their detailed comments which helped to improve the quality of the paper.

References

[1] N. Belkin, W.B. Croft, Information filtering and information retrieval: Two sides of the same coin? Communications of the ACM 35 (1992) 29–38.
[2] G. Bordogna, G. Pasi, A flexible multi criteria information filtering model, Soft Computing 14 (2010) 799–809.
[3] M. Boughanem, C. Chrisment, L. Tamine, Genetic approach to query space exploration, Information Retrieval 1 (1999) 175–192.
[4] M. Boughanem, C. Chrisment, L. Tamine, On using genetic algorithms for multimodal relevance optimization in information retrieval, Journal of the American Society for Information Science and Technology 53 (2002) 934–942.
[5] M. Boughanem, H. Tebri, M. Tmar, IRIT at TREC'2002: Filtering track, in: Proceedings of the 11th Text REtrieval Conference, 2002, pp. 337–344.
[6] N. Cancedda, N. Cesa-Bianchi, A. Conconi, C. Gentile, C. Goutte, T. Graepel, Y. Li, J.M. Renders, J. Shawe-Taylor, A. Vinokourov, Kernel methods for document filtering, in: Proceedings of the 11th Text REtrieval Conference, 2003, pp. 373–382.
[7] R. Cecchini, C. Lorenzetti, A. Maguitman, Using genetic algorithms to evolve a population of topical queries, Information Processing and Management 44 (2008) 1863–1878.
[8] C.C. Chang, C.J. Lin, LIBSVM: A library for support vector machines, Technical report, http://www.csie.ntu.edu.tw/~cjlin/libsvm/, 2011.
[9] H. Chen, G. Shankaranarayanan, L. She, A. Iyer, A machine learning approach to inductive query by example: An experiment using relevance feedback, ID3, genetic algorithms, and simulated annealing, Journal of the American Society for Information Science 49 (1998) 693–705.
[10] R.L. Cilibrasi, P.M.B. Vitanyi, The Google similarity distance, IEEE Transactions on Knowledge and Data Engineering 19 (2007) 370–383.
[11] K. Collins-Thompson, P. Ogilvie, Y. Zhang, J. Callan, Information filtering, novelty detection, and named-page finding, in: Proceedings of the 11th Text REtrieval Conference, 2002, pp. 107–140.
[12] O. Cordón, E. Herrera-Viedma, C. López-Pujalte, M. Luque, C. Zarco, A review on the application of evolutionary computation to information retrieval, International Journal of Approximate Reasoning 34 (2003) 241–264.
[13] O. Cordón, E. Herrera-Viedma, M. Luque, Improving the learning of Boolean queries by means of a multiobjective IQBE evolutionary algorithm, Information Processing and Management 42 (2006) 615–632.
[14] O. Egozi, E. Gabrilovich, S. Markovitch, Concept-based feature generation and selection for information retrieval, in: Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, 2008, pp. 1132–1137.
[15] J.R. Finkel, T. Grenader, C. Manning, Incorporating non-local information into information extraction systems by Gibbs sampling, in: Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics, 2005, pp. 363–370.
[16] L. Finkelstein, E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman, E. Ruppin, Placing search in context: The concept revisited, ACM Transactions on Information Systems 20 (2002) 116–131.
[17] A. Fonseca-Bruzón, R. Gil-García, A. Pons-Porrata, Using the $\alpha\beta$-neighborhood for adaptive document filtering, in: Proceedings of the 13th Iberoamerican Congress on Pattern Recognition: Progress in Pattern Recognition, Image Analysis and Applications, 2008, pp. 783–790.
[18] E. Gabrilovich, S. Markovitch, Overcoming the brittleness bottleneck using Wikipedia, in: Proceedings of the 21st National Conference on Artificial Intelligence, 2006, pp. 1301–1306.
[19] E. Gabrilovich, S. Markovitch, Computing semantic relatedness using Wikipedia-based explicit semantic analysis, in: Proceedings of the 20th International Joint Conference on Artificial Intelligence, 2007, pp. 1606–1611.
[20] E. Gabrilovich, S. Markovitch, Wikipedia-based semantic interpretation for natural language processing, Journal of Artificial Intelligence Research 34 (2010) 443–498.
[21] S. García, F. Herrera, An extension on "Statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons, Journal of Machine Learning Research 9 (2008) 2677–2694.
[22] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. Witten, The WEKA data mining software: An update, ACM Special Interest Group on Knowledge Discovery and Data Mining Explorations Newsletter 11 (2009) 10–18.
[23] J. Hendler, E. Feigenbaum, Knowledge is power: The semantic web vision, in: Web Intelligence: Research and Development, Springer, Berlin, 2001, pp. 18–29.
[24] M. Hepp, D. Bachlechner, K. Siorpaes, Harvesting Wiki consensus – Using Wikipedia entries as ontology elements, in: IEEE Internet Computing, 2006, pp. 54–65.
[25] J.T. Horng, C.C. Yeh, Applying genetic algorithms to query optimization in document retrieval, Information Processing and Management 36 (2000) 737–759.
[26] J. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection, MIT Press, 1992.

[27] D. Kraft, F. Petry, B. Buckles, T. Sadasivan, Genetic algorithms for query optimization in information retrieval: relevance feedback, in: E. Sanchez, T. Shibata, L. Zadeh (Eds.), Genetic Algorithms and Fuzzy Logic Systems Soft Computing Perspectives, 1995, pp. 155–173.
[28] J. Lafferty, A. McCallum, F. Pereira, Conditional random fields: Probabilistic models for segmenting and labeling sequence data, in: Proceedings of the 18th International Conference on Machine Learning, 2001, pp. 282–289.
[29] A. López-Herrera, E. Herrera-Viedma, F. Herrera, A study of the use of multi-objective evolutionary algorithms to learn boolean queries: A comparative study, Journal of the American Society for Information Science and Technology 60 (2009) 1192–1207.
[30] A. López-Herrera, E. Herrera-Viedma, F. Herrera, Applying multi-objective evolutionary algorithms to the automatic learning of extended Boolean queries in fuzzy ordinal linguistic information retrieval systems, Fuzzy Sets and Systems 160 (2009) 2192–2205.
[31] P. Malo, P.A. Siitari, O. Ahlgren, J. Wallenius, P. Korhonen, Semantic content filtering with Wikipedia and ontologies, in: Proceedings of IEEE International Conference on Data Mining Workshops, 2010, pp. 518–526.
[32] M. McHale, A Comparison of WordNet and Roget's taxonomy for measuring semantic similarity, in: Proceedings of COLING/ACL Workshop on Usage of WordNet in Natural Language Processing Systems, 1998, pp. 115–120.
[33] O. Medelyan, D. Milne, Augmenting domain-specific thesauri with knowledge from Wikipedia, in: Proceedings of the New Zealand Computer Science Research Student Conference, 2008, pp. 108–114.
[34] O. Medelyan, D. Milne, C. Legg, I. Witten, Mining meaning from Wikipedia, International Journal of Human–Computer Studies 67 (2009) 716–754.
[35] O. Medelyan, I. Witten, D. Milne, Topic indexing with Wikipedia, in: Proceedings of the First AAAI Workshop on Wikipedia and Artificial Intelligence, 2008, pp. 19–24.
[36] K. Meffert, JGAP – Java genetic algorithms and genetic programming package, Technical report, http://jgap.sf.net, 2010.
[37] E. Meij, M. Bron, L. Hollink, B. Huurnink, M. de Rijke, Learning semantic query suggestions, in: Proceedings of the 8th International Semantic Web Conference, 2009, pp. 415–430.
[38] R. Mihalcea, A. Csomai, Wikify!: linking documents to encyclopedic knowledge, in: Proceedings of the 16th ACM Conference on Information and Knowledge Management, 2007, pp. 233–242.
[39] D. Milne, Computing semantic relatedness using Wikipedia link structure, in: Proceedings of the New Zealand Computer Science Research Student Conference, 2007.
[40] D. Milne, I. Witten, Learning to link with Wikipedia, in: Proceedings of the 17th ACM Conference on Information and Knowledge Management, 2008, pp. 509–518.
[41] D. Milne, I. Witten, An open-source toolkit for mining Wikipedia, in: Proceedings of the New Zealand Computer Science Research Student Conference, 2009.
[42] D. Milne, I. Witten, D. Nichols, A knowledge-based search engine powered by Wikipedia, in: Proceedings of the 16th ACM Conference on Information and Knowledge Management, 2007, pp. 445–454.
[43] V. Nastase, M. Strube, Decoding Wikipedia categories for knowledge acquisition, in: Proceedings of the 23rd AAAI Conference on Artificial Intelligence, 2008, pp. 1219–1224.
[44] V. Nastase, M. Strube, B. Bórschinger, C. Zirn, A. Elghafari, WikiNet: A very large scale multi-lingual concept network, in: Proceedings of the 7th Conference on International Language Resources and Evaluation, 2010, pp. 1015–1022.
[45] R. Navigli, G. Crisafulli, Inducing word senses to improve web search result clustering, in: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2010, pp. 116–126.
[46] R. Navigli, S. Ponzetto, Babelnet: building a very large multilingual semantic network, in: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, 2010, pp. 216–225.
[47] J. Nothman, J. Curran, T. Murphy, Transforming Wikipedia into named entity training data, in: Australasian Language Technology Association Workshop, 2008, pp. 124–132.
[48] G. Pasi, G. Bordogna, R. Villa, A multi-criteria content-based filtering system, in: Proceedings of the 30th Annual International ACM Special Interest Group on Information Retrieval Conference on Research and Development in Information Retrieval, ACM, 2007, pp. 775–776.
[49] R. Poli, W. Langdon, N. McPhee, A field guide to genetic programming, published via http://lulu.com and freely available at http://www.gp-field-guide.org.uk, 2008.
[50] S. Ponzetto, R. Navigli, Large-scale taxonomy mapping for restructuring and integrating Wikipedia, in: Proceedings of the 21st International Joint Conference on Artificial Intelligence, 2009, pp. 2083–2088.
[51] S. Ponzetto, M. Strube, Exploiting semantic role labeling, WordNet and Wikipedia, in: Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL, 2006, pp. 192–199.
[52] S. Ponzetto, M. Strube, An API for measuring the relatedness of words in Wikipedia, in: Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions, 2007, pp. 49–52.
[53] S. Ponzetto, M. Strube, Knowledge derived from Wikipedia for computing semantic relatedness, Journal of Artificial Intelligence Research 30 (2007) 181–212.
[54] J. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann, 1993.
[55] C. van Rijsbergen, Information Retrieval, Butterworths, 1979.
[56] S. Robertson, I. Soboroff, The TREC 2002 filtering track report, in: Proceedings of the 11th Text REtrieval Conference, 2002, pp. 439–452.
[57] J. Rocchio, Relevance feedback in information retrieval, in: G. Salton (Ed.), The SMART Retrieval System, Prentice-Hall, 1971, pp. 313–323.
[58] G. Salton, C. Buckley, Improving retrieval performance by relevance feedback, Journal of the American Society for Information Science 41 (1990) 288–297.
[59] M. Smith, M. Smith, The use of genetic programming to build boolean queries for text retrieval through relevance feedback, Journal of Information Science 23 (1997) 423–431.
[60] J. Sowa, The challenge of knowledge soup, in: J. Ramadas, S. Chunawala (Eds.), Research Trends in Science, Technology and Mathematics Education, 2004, pp. 55–90.
[61] M. Strube, S. Ponzetto, WikiRelate! Computing semantic relatedness using Wikipedia, in: Proceedings of the 21st AAAI Conference on Artificial Intelligence, 2006, pp. 1419–1424.
[62] F. Suchanek, G. Kasneci, G. Weikum, Yago: A large ontology from Wikipedia and WordNet, Elsevier Journal of Web Semantics 6 (2008) 203–217.
[63] L. Tamine, C. Chrisment, M. Boughanem, Multiple query evaluation based on an enhanced genetic algorithm, Information Processing and Management 39 (2003) 215–231.
[64] E. Voorhees, L. Buckland, NIST special publication: SP 500-251, in: The Eleventh Text Retrieval Conference, published online at http://comminfo.rutgers.edu/~muresan/IR/TREC/Proceedings/t11_proceedings/t11_proceedings.html, 2002.
[65] H. Xu, Z. Yang, B. Wang, B. Liu, J. Cheng, Y. Liu, Z. Yang, X. Cheng, S. Bai, TREC-11 experiments at CAS-ICT: Filtering and Web, in: Proceedings of the 11th Text REtrieval Conference, 2002, pp. 141–165.
[66] J.J. Yang, R. Korfhage, Query modification using genetic algorithms in vector space models, International Journal of Expert Systems Research and Applications 7 (1994) 165–191.
[67] H. Zhuge, Interactive semantics, Artificial Intelligence 174 (2010) 190–204.