Galitsky B., Kovalerchuk B., Improving Web Search Relevance with Learning Structure of Domain Concepts, In: Clusters, Orders, and Trees: Methods and Applications, Aleskerov, F. Goldengorin, B. Pardalos, P.M. (Eds.), Springer, 2014, pp. 311-363.

Improving Web Search Relevance with Learning Structure of Domain Concepts

Boris A. Galitsky

eBay Inc. San Jose CA, USA

boris.galitsky@ebay.com

Boris Kovalerchuk

Central Washington University,

Ellensburg, WA, USA

Boris.Kovalerchuk@cwu.edu

Abstract. This paper addresses the problem of improving the relevance of a search engine results in a vertical domain. The proposed algorithm is built on a structured taxonomy of keywords. The taxonomy construction process starts from the seed terms (keywords) and mines the available source domains for new terms associated with these entities. These new terms are formed in several steps. First the snippets of answers generated by the search engine are parsed producing parsing trees. Then commonalities of these parsing trees are found by using a machine learning algorithm. These commonality expressions then form new keywords as parameters of existing keywords, and are turned into new seeds at the next learning iteration. To match NL expressions between source and target domains, the proposed algorithm uses syntactic generalization, an operation which finds a set of maximal common sub-trees of constituency parse trees of these expressions.

The evaluation study of the proposed method revealed the improvement of search relevance in vertical and horizontal domains. It had shown significant contribution of the learned taxonomy in a vertical domain, and a noticeable contribution of a hybrid system (that combines of taxonomy and syntactic generalization) in the horizontal domains. The industrial evaluation of a hybrid system reveals that the proposed algorithm is suitable for integration into industrial systems. The algorithm is implemented as a component of Apache OpenNLP project.

Keywords: learning taxonomy, learning syntactic parse tree, transfer learning, syntactic generalization, search relevance

1. Introduction

The goal of this paper is improving relevance of web search by adding specific types of semantic information. Consider a web search task with query Q and answers

a_1, a_2,…a_n that are produced by some conventional search engine. Assume that an *automatic semantic analysis algorithm* F is applied to the query question Q and produced a statement "Q *is about* X", e.g., "query Q is about Tax". We will denote such statement as S(Q,X), where S can be viewed as a predicate "is_about(,). Thus, F(Q)=S(Q,X).

Similarly assume that the algorithm F is applied to each answer a_i and produced statements: "a_1 is about X_1", "a_2 is about X_2",…, "a_i is about X_i",…, "a_n is about X_n". Thus, we have $F(a_i)=S(a_i,X_i)$ for all answers. Let also L be a *score function* that measures the *similarity* between pairs $<Q,S(Q,X)>$ and $<a_i,S(a_i,X_i)>$, $L(<Q,S(Q,X)>, <a_i,S(a_i,X_i)>)$ as a mapping to the interval [0,1]. Then we can re-rank answers a_i obtained by a conventional search engine, relative to L value. The answers with the highest scores

$$\max_{i=1,2\ldots n} L(<Q,S(Q,X)>, <a_i,S(a_i,X_i)>)$$

are considered as the *most relevant*.

The proposed measure of similarity takes into account the traditional approach, which takes all keywords from questions and answers, with our specific method of matching the keywords we determine as being essential. The above similarity will be assessed via mapping both $<Q,S(Q,X)>$ and $<a_i,S(a_i,X_i)>$ into a constructed structure which we refer to as *taxonomy*. Instead of keywords or bag-of-words approaches, we will also be computing similarity between parse trees for questions and answers.

This paper elaborates this approach and is organized as follows. We start from the design of the analysis algorithm F that performs the taxonomy-supported search, then we design the similarity measure L to further improve search relevance. After that we demonstrate the efficiency of the proposed method on real web searches. The paper concludes with a detailed analysis of the related works, advantages of the proposed method, and expected future work.

The algorithm uses a structured taxonomy of keywords. The taxonomy construction process starts from the seed terms (keywords) and mines the available source domains for new terms associated with these terms. These new terms are formed in several steps that involve the search engine results, parsing trees and a machine learning algorithm. To match NL expressions between source and target domains, the algorithm uses syntactic generalization, an operation which finds a set of maximal common subtrees of constituency parse trees of these expressions.

The industrial evaluation of a hybrid system reveals that the proposed algorithm is suitable for integration into industrial systems. The algorithm is implemented as a component of Apache OpenNLP project.

2. Method

2.1. Defining is_about relation

The is_about relation helps to 'understand' the *root concepts* of the query Q and obtain the best answer a_i. In other words, this relation sets up the set of essential keywords of keywords for Q or a_i. Let K(Q) be a set of keywords of Q (the function that extracts meaningful keywords from a sentence, depending on the current choice of stop-words). S(Q,X) is defined as *is_about*(K(Q), X),

$$S(Q,X) = \textit{is\_about}(K(Q), X),$$

where X is a subset of K(Q). Similarly, for the answer a_i, we have $S(a_i,X_i)$ = *is_about*($K(a_i)$, X).

Let query Q be about *b* and K(Q)={*a b c*}, i.e., *is_about* ({*a,b,c*}, *b*). Then we understand that other queries with {*a b*} and {*b c*} are relevant or marginally relevant to query Q, and a query with keywords {*a c*} is irrelevant to Q. In other words, b is an essential in Q and the other query without *b* term is meaningless relative to Q, and an answer which does not contain b is *irrelevant* to the query which includes *b* .

Example1. Let the set of keywords {*computer, vision, technology*}, {*computer, vision*}, {*vision, technology*} be relevant to the query Q, and {*computer, technology*} are not, thus the query Q is about {*vision*}.

Notice that for keywords in the form of a *noun phrase* or a *verb phrase* the head or a verb may not be a keyword. Also we can group words into phrases when they form an entity, e.g., bill-gates: *is_about*({*vision, bill-gates, in- computing*}, {*bill-gates*}).

A set of keywords as called *essential* if it occurs on the right side of *is_about. is_about* relation as a *relation between a set of keywords and its ordered subset*.

Example2. Let *b* be essential for Q with K(Q)={*a, b, c, d*}, (*is_about*({*a, b, c, d*}, {*b*}) and *c* also be essential when *b* is in the query, *is_about*({*a, b, c, d*}, {*b,c*}. Here *b* and *c* are ordered with *b* being more essential then *c*. Then queries and answers with {*a, b, c*}, {*b, c, d*}, {*b, c*} are considered to be relevant to Q because they contain both essential keywords. In contrast, queries and answers with {*a, b*}, {*b, d*} are considered to be (marginally) relevant to Q because *c* is missed and they are likely less specific. Accordingly queries and answers with {*a, d*} only are considered to be irrelevant to Q.

This example gives an *idea* that we use to define the *relevance similarity score* L between the query and its answers. It is based on the order of how essential are the keywords, not only on a simple overlap of keywords or essential keywords between Q and a_i. Hence, for a query Q with K(Q)= {*a, b, c, d*} and two answers (snippets) with $K(a_1)$={*b, c, d, ...,e, f, g*} and $K(a_2)$={*a, c, d, ..., e, f, g*}, the former is relevant and the latter is not because *c* to be essential requires such keyword *b* that has a higher rank of essentiality than *c* for the query Q with ordered essential keywords {*b,c*}.

Above we defined it using an *ordered set of essential keywords.* This definition works for two essential keywords. For more than two keywords we may have *multiple essentiality orders*, e.g., for three essential keywords {*b,c,d*} we can have ordered sets {*b,c*} and {*b, d*} without {*b,c,d*} that *c* is not required for *d* to be essential. To encode

this essentiality order, we need to employ a tree structure, which is going to be a taxonomy. For a query, we build a structure of its keywords with essentiality relations on them by the function K, introduced above. For two keywords, one is more essential than the other if there is a path in the taxonomy tree (starting from the root) which includes the nodes with these keywords as labels, and the first node is between the second node and the root.

2.2. Defining essential keywords

Definition: A set of keywords E as called *essential* if it occurs on the right side of is_about relation and E is a *structured subset* of the set of keywords, that is the *partial* order essentiality relation "$<_E$" is defined for every pair of elements of E, $\langle E, <_E \rangle$. In this paper we limit the essentiality structure to a *tree structure* that we call a *keyword taxonomy*.

Now we need to define a method to construct essential keywords E for the query Q. Assume that all queries are from a vertical domain, e.g., tax domain. Then we can build a *common tree of concepts for the domain* (domain *taxonomy*) T, e.g., for tax domain. Later on in this paper we will describe the method for building T. The tree T includes paths which correspond with typical queries. We rely on an assumption for a vertical domain that for two words in a given domain, one word is always more essential than the other for all queries. Based on this assumption, a single taxonomy can support a search in the whole vertical domain.

Having T we can define E by set-intersection T as a set and Q, $E = T \cap Q$. This gives us the relation cover is_about(K(Q), E). Thus one of the ways to define a query analysis algorithms F that produces *is_about* relation, from Q using T, F(Q)=*is_about*(Q, $T \cap Q$). Alternative ways to get E is to intersect keywords K(Q) of Q with T, $E= T \cap K(Q)$ or to intersect Q with an individual path T_p in T, getting $E= T_p \cap Q$ and is_about(Q, $T_p \cap Q$)

We say that a path T_p covers a query Q if the set of keywords for the nodes of T_p is a super-set of Q. If multiple paths cover a query Q producing different intersections $Q \cap T_p$ then this query has multiple meanings in the domain; for each such meaning a separate set of acceptable answers is expected.

The search based on such tree structures is typically referred to as the *taxonomy-based search* (Galitsky 2003). It identifies terms that *should* occur in the answer and terms that *must* occur there, otherwise the search result is irrelevant. The keywords that should occur are from the taxonomy T, but the keywords that must occur are from both the query Q and taxonomy T. This is a totally different mechanism than a conventional TF*IDF based search (Manning et al 2008)

2.3. Constructing relevance score function L

The introduction of the essentiality order for keywords leads to the specification of the relevance score function $L(<Q,S(Q,X)>,<a_i,S(a_i,X_i)>)$, where now X and X_i are

trees of essential keywords derived by the function K. Below we discuss a method to build this function.

Consider a situation where all essential words X from the query Q are also present in the answer, $X \subseteq X_i$ the answer a_i is called *partially acceptable* for query Q. This can be defined as:

$$\text{If } X \subseteq X_i \text{ then } L(<Q,S(Q,X)>,<a_i,S(a_i,X_i)>)=1.$$

However, this is a semantically shallow method especially for common situations where X and X_i contains only few words. A better way is use a *common tree of concepts for the domain* T and use it to measure similarity between X and X_i. Now we can define L given T for the acceptable case,

$$\text{If } X \subseteq X_i \subseteq T \text{ then } L(<Q,S(Q,X)>,<a_i,S(a_i,X_i)>)=1.$$

This means that tree X is a subtree of tree X_i and both trees are subtrees of tree T.

The requirement of weak acceptability, that X is a subtree of T, is desirable, but very restrictive. Therefore, we define acceptability in the following way. An answer $a_i \in A$ is *acceptable* if it includes *all essential* (according to *is_about*) keywords from the query *Q* as found in the taxonomy path $T_p \in T$. For any taxonomy path T_p which covers the question *q* (intersections of their keywords is not empty), these intersection keywords **must be** in the acceptable answer a_i.

$$\forall\, T_p \in T:\ T_p \cap X \neq \varnothing \Rightarrow X_i \supseteq T_p \cap X.$$

In other words, *X* as a set/tree of keywords for a question, which are essential in this domain (covered by a path in the taxonomy), must be a subset of X_i (the set/tree of keywords for this answer). This is a more complex (but not necessarily stronger) requirement than $X \subseteq X_i$. $\in$ used here to denote a path in a tree.

For the best answer (most accurate) we write

$$a_{best}:\ max_i\ (|X_i \cap (T_p \cap X)|),\ T_p \in T.$$

Accordingly, we define **a taxonomy-based relevance score** L as the value of *cardinality* $|X_i \cap (T_p \cap X)|$, computed for all T_p which cover *Q*. Then the best answer is found among the scores for all answers *A*. The score L can be normalized by dividing it by |X| to get it in [0,1] interval.

The taxonomy-based score can be combined with the other scores such as TF*IDF, temporal/decay parameter, location distance, pricing, linguistic similarity, and other scores for the resultant ranking, depending on search engine architecture. In our evaluation we will be combining it with the linguistic similarity score (Section 2.6). Hence L is indeed depends not only on X and X_i but also on Q and a_i.

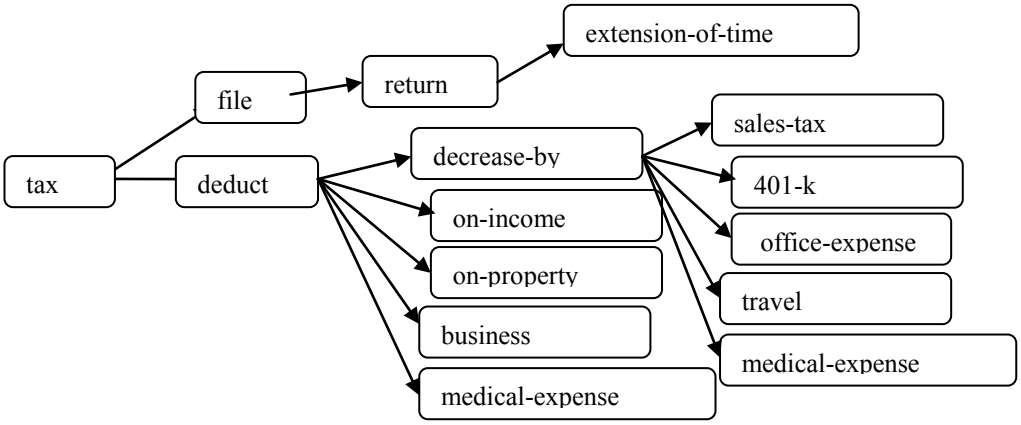


Fig.1: An example of a snapshot of a domain taxonomy

Example. Consider a taxonomy T presented in Fig. 1 for the query Q =‘*How can tax deduction be decreased by ignoring office expense’,* with a set of keywords K(Q)={*how, can, tax, deduct(ion), decreas(ed)-by, ignor(ing), office, expense*} and a set of tree answers A={a_1,a_2,a_3} presented as keywords:
a_1= {*deduct, tax, business, expense, while, decreas(ing), holiday, travel, away, from, office*},
a_2 = {*pay, decreas(ed), sales-tax, return, trip, from, office, to, holiday, no, deduct(ion)*},
a_3 = {*when, file, tax, return, deduct, decrease-by, not, calculate, office, expense, and, employee, expense*}}.
Notice that a_2 includes the multiword *sales*-tax from the taxonomy, which is also counted as a set of two words {*sales, tax*}. However, in a_3 *decrease-by* is considered as a single word because our function K considers prepositions as stop words and does not count them separately.
We show ending in brackets for convenience, omitting tokenization and word form normalization. In terms of keyword overlap, a_1, a_2 and a_3 all look like good answers.

In accordance with given T query Q is covered by the path T_p = {<tax> - <deduct> - < decrease-by>-<office-expense>}. We calculate the similarity score for each answer with Q:

score(a_1) =*cardinality*($a_1 \cap (T_p \cap Q)$) = *cardinality*({*tax, deduct*}) = 2;
score(a_2) = cardinality({*tax, deduct*}) = 2;
score(a_3) = cardinality({*tax, deduct, decrease-by, office-expense*}) = 3.;

The answer a_3 is the best answer in this example. Our next example is about disambiguation of keywords.

Example. Consider a query q= "*When can I file extension of time for my tax return*?" with two answers:

a_1 = "You need to file form 1234 to request a 4 month extension of time to file your tax return"

a_2= "You need to download file with extension 'pdf', print and complete it to do your taxes"
and the closest taxonomy path: T_p = {<*tax*> -< *file*>-<*return*> -< *extension-of-time*>}. In this example both a_1 and a_2 contain word extension, but the keyword is "extension-of-time" not *extension*. Resolving this ambiguity leads to a higher score for a_1.

2.4. Taxonomy-based relevance verification algorithm

We now outline the algorithm, which takes a query Q, runs a search (outside of this algorithm), gets a set of candidate answers A and finds the best acceptable answer according to the definitions given above.

The input: query Q
The output: the best answer a_{best} and the set of acceptable answers A_a

---

1) For a query Q, obtain a set of candidate answers A by available means (using keywords, using internal index, or using external index of search engine's APIs);
2) Find a path of taxonomy T_p which covers maximal number of terms in Q, along with other paths, which cover Q, to form a set P = { T_{p1}, T_{p2}, …}.
   Unless acceptable answer is found:
3) Compute the set $T_p \cap Q$.
   For each answer $a_i \in A$
4) Compute $a_i \cap (T_p \cap Q)$) and test if all essential words from the query, which exists in T_p are also in the answer (acceptability test)
5) Compute similarity score of Q with or each a_i
6) Compute the best answer a_{best} and the set of acceptable answers A_a.
   If no acceptable answer found, return to 2 for the next path from P.
7) Return a_{best} and the set of acceptable answers A_a if available.

This algorithm filters out irrelevant answers by searching for taxonomy path (down to a leaf node if possible) which is closest to the given query in terms of the number of entities from this query. Then this path and leave node specify most accurate meaning of the query, and constrain which entities *must* occur and which *should* occur in the answer to be considered relevant. If the *n*-th node entity from the question occurs in answer, then all $k < n$ entities should occur in it as well.

2.5. Building taxonomy

The domain tree/taxonomy is built iteratively. The algorithm starts from a "taxonomy seed" that contains at least 2-3 initial nodes (keywords) including the root of the tree. Each next iteration step *k* adds edges and nodes to specify existing nodes known at the

step *k*-1. A seed taxonomy can be made manually. An alternative way is using external sources, such as a glossary of that knowledge domain, e.g., http://www.investopedia.com/categories/taxes.asp for tax domain.

Example. The seed for the tax domain can contain *tax* as a root of the tree (a domain-determining entity), and {*deduct, income, property*} as nodes of the next level of the tree that are main entities in this domain. Another option for the seed taxonomy is presented in Fig 1 with tax as a root, and file and deduct as child nodes at the next level.

Each iteration step is accomplished as a *learning* step using *web mining* to learn next nodes such as *sales-tax*, *401k*, etc (Fig. 1).

The learning algorithm:

(1) takes a pair (root node, child node), such as (*tax, deduct),*

(2) use it as a search query via a search engine, e.g., Bing,

(3) extracts words and expressions which are *common* among search results (Section 2.6),

(4) expands the tree with these common words as new terminal nodes,

(5) takes a triple of nodes (node, child, grandchild) and repeat steps (2)-(4) for this triple.

These process can continue for k-tuples (paths on the tree) with k>3 to get a deeper domain taxonomy. Here common words are single verbs, nouns, adjectives and even adverbs, prepositional phrases or multi-words, including propositional, noun and verb phrases, which occur in *multiple* search results. The details of the extraction of common expressions between search results are explained later.

Example. Fig. 2 shows some search results on Bing.com for the tree path tax-deduct-decrease. E.g. for the path *tax - deduct* newly learned entities can be

tax - deduct → decrease-by
tax - deduct → property-of
tax - deduct → medical-expense.
tax - deduct → of-income
tax - deduct → business

The format here is *existing_entity → new_entity*, *'→'* here is an unlabeled edge of the taxonomy extension at the current learning step.

Next we run triples getting:

tax-deduct-decrease-by → sales
tax-deduct-decrease → sales-tax
tax-deduct-decrease-by →401-K
tax-deduct-decrease-by → medical
tax-deduct - of-income → rental
tax-deduct - of-income → itemized
tax-deduct - of-income → mutual-funds

How to **Decrease** Your Federal Income **Tax** | eHow.com
the Amount of Federal **Taxes** Being Withheld; How to Calculate a Mortgage Rate After Income **Taxes**; How to **Deduct** *Sales **Tax*** From the Federal Income **Tax**

Itemizers Can **Deduct** Certain **Taxes**
... may be able to **deduct** certain **taxes** on your federal income **tax** return? You can take these **deductions** if you file Form 1040 and itemize **deductions** on Schedule A. **Deductions decrease** ...

Self Employment Irs Income **Tax** Rate Information & Help 2008, 2009 ...
You can now **deduct** up to 50% of what has been paid in self employment **tax**. · You are able to **decrease** your self employment income by 7.65% before figuring your **tax** rate.

How to Claim Sales **Tax** | eHow.com
This amount, along with your other itemized **deductions**, will **decrease** your taxable ... How to **Deduct** Sales **Tax** From Federal **Taxes**; How to Write Off *Sales* **Tax**; Filling **Taxes** with ...

Prepaid expenses and **Taxes**
How would prepaid expenses be accounted for in determining **taxes** and accounting for ... as the cash effect is not yet determined in the net income, and we should **deduct** a **decrease**, and ...

How to **Deduct** *Sales* **Tax** for New Car Purchases: Buy a New Car in ...
How to **Deduct** Sales **Tax** for New Car Purchases Buy a New Car in 2009? Eligibility Requirements ... time homebuyer credit and home improvement credits) that are available to **decrease** the ...

Fig.2: Search results on Bing.com for the current taxonomy tree path *tax-deduct-decrease.*

We outline the iterative algorithm, which takes a taxonomy with its terminal nodes, and attempts to extend them via web mining to acquire a new set of terminal nodes. At the iteration k we acquire a set of nodes, extending current terminal node t_i with t_{ik1} , t_{ik2} .... This algorithm is based on the operation of generalization, which takes two texts as sequences <*lemma(word), part-of-speech*> and gives least general set of texts in this form (Section 2.6). We outline the iterative step:

Input: Taxonomy T_k with terminal nodes $\{t_1, t_2 \ldots t_n\}$

A threshold for the number of occurrences to provide sufficient evidence for inclusion into T_k: *th(k, T).*

Output: extended taxonomy T_{k+1} with terminal nodes
$\{t_{1k1}, t_{1k2}, \ldots, t_{2k1}, t_{2k2}, \ldots t_{nk1}, t_{nk2}\}$

For each terminal node t_i

1) Form a search query as a path from the root to t_i , q = $\{ t_{root}, \ldots, t_i\}$;
2) Run web search for q and get a set of answers (snippets) A_q.
3) Compute a pair-wise generalization (Section 2.6) for answers A_q :
   $\Lambda(A_q) = a_1 \wedge a_2, a_1 \wedge a_3, \ldots, a_1 \wedge a_m, \ldots, \ldots, a_{m-1} \wedge a_m$,
4) Sort all elements (words, phrases) of $\Lambda(A_q)$ in descending order of the number of occurrences in $\Lambda(A_q)$. Retain only the elements of $\Lambda(A_q)$ with the number of occurrences above a threshold *th(k, T).* We call this set $\Lambda^{high}(A_q)$.
5) Subtract the labels from all existing taxonomy nodes from $\Lambda^{high}(A_q)$: $\Lambda^{new}(A_q) = \Lambda^{high}(A_q) / T_{k..}$ We maintain the uniqueness of labels of taxonomy to simplify the online matching algorithm.
6) For each element of $\Lambda^{high}(A_q)$, create a taxonomy node $t_{ihk,}$ where $h \in \Lambda^{high}(A_q)$, and k is the current iteration number, and add the taxonomy edge $(t_{i,}, t_{ihk,})$ to $T_{k.}$

The default value of *th(k, T)* is 2. However there is an empirical limit on how many nodes are added to a given terminal node at each iteration. This limit is 5 nodes per iteration, so we take the five highest numbers of occurrences of a term in distinct search results. This constraint helps to maintain the tree topology for the taxonomy being learned. The resultant taxonomy is a tree which is neither binary nor sorted or balanced.

Given the algorithm for the iteration step, we apply it to the set of main entities at the first step, to build the whole taxonomy:

Input: Taxonomy T_o with nodes $\{t_1, t_2 \ldots t_n\}$ which are main entities
Output: resultant taxonomy T with terminal nodes

Iterate through k:
  Apply iterative step to k
  If T_{k+1} has an empty set of nodes to add, stop

2.6. Algorithm to extract common words/expressions among search results

The word extraction algorithm is applied to a pair of sentences from two search results. These sentences are viewed as parsing trees. The algorithm produces a set of maximal common parsing sub-trees that constitute structured set of common words in these sentences. We refer the reader to further details in (Galitsky et al 2012, Galitsky 2013).

Input: a pair of sentences
Output: a set of maximal common sub-trees

---

1) Obtain the parsing tree for each sentence, using OpenNLP. For each word (tree node), we have a lemma, a part of speech and the form of the word's information. This information is contained in the node label. We also have an arc to the other node.
2) Split sentences into sub-trees that are phrases for each type: verb, noun, prepositional and others. These sub-trees are overlapping. The sub-trees are coded so that the information about their occurrence in the full tree is retained.
3) All the sub-trees are grouped by phrase types.
4) Extend the list of phrases by adding equivalence transformations (Galitsky et al 2010).
5) Generalize each pair of sub-trees for both sentences for each phrase type.
6) For each pair of sub-trees, yield an alignment (Gildea 2003), and generalize each node for this alignment. Calculate the score for the obtained set of trees (generalization results).
7) For each pair of sub-trees of phrases, select the set of generalizations with the highest score (the least general).

8) Form the sets of generalizations for each phrase type whose elements are the sets of generalizations for that type.
9) Filter the list of generalization results: for the list of generalizations for each phrase type, exclude more general elements from the lists of generalization for a given pair of phrases.

For a given pair of words, only a single generalization exists; if words are the same in the same form, the result is a node with this word in this form. We refer to the generalization of words occurring in a syntactic tree as a *word node*. If the word forms are different (e.g., one is single and the other is plural), only the lemma of the word remains. If the words are different and only the parts of speech are the same, the resultant node contains only the part-of-speech information with no lemma. If the parts of speech are different, the generalization node is empty.

For a pair of phrases, the generalization includes all the *maximum* ordered sets of generalization nodes for the words in the phrases so that the order of words is retained. Consider the following example:

To buy the digital camera today, on Monday
The digital camera was a good buy today, the first Monday of the month

The generalization is {<*JJ-digital, NN-camera*>, <*NN-today, ADV, NN-Monday*>}, where the generalization for the noun phrase is followed by the generalization for the adverbial phrase. The verb *buy* is excluded from both generalizations because it occurs in a different order in the above phrases. *Buy - digital - camera* is not a generalization phrase because *buy* occurs in a different sequence in the other generalization nodes.

We can see that the multiple maximum generalizations occur depending on how the correspondence between words is established; multiple generalizations are possible. Ordinarily, the total of the generalizations forms a lattice. To obey the condition of the maximum, we introduce a score for generalization. The scoring weights of generalizations are decreasing, roughly, in the following order: nouns and verbs, other parts of speech, and nodes with no lemma, only a part of speech. In its style, the generalization operation follows the notion of the 'least-general generalization' or anti-unification, if a node is a formula in a language of logic. Therefore, we can refer to the syntactic tree generalization as an operation of *anti-unification of syntactic trees*.

3. Results on improving web search relevance

3.1 An example of taxonomy learning session

Let $G = \{g_i\}$ be a fixed set of linguistic relations between the pairs of terms. For instance, we may have a relation g_i(tax deduction, reduces). In this relation 'reduces' serves as an *attribute* of 'tax deduction' term. An attribute of the term t that occurs in more than one answer is called a *parameter* of t.

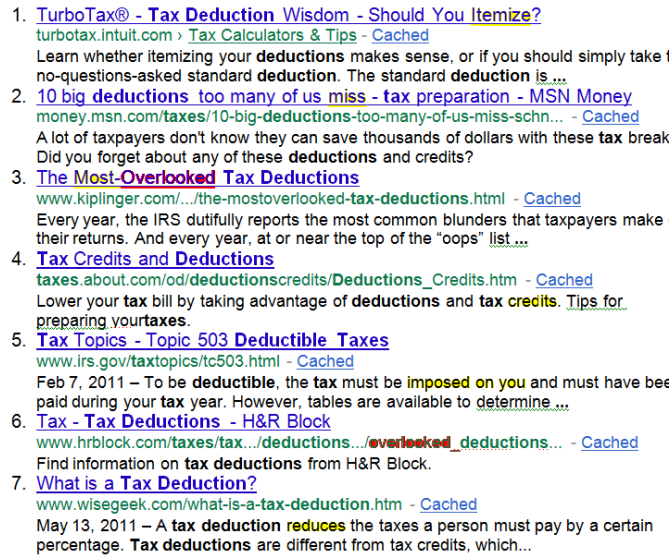
Consider a seed taxonomy as a pair (*tax-deduct*). The taxonomy learning session consists of the following steps:

1) Getting search results A ={a_i} for the pair (*tax-deduct*) using some web search engine.

2) Finding in each search results A_i the terms that are candidate attributes of 'tax' and/or 'deduct' (highlighted in Fig. 3a).

3) Turning candidate attributes into parameters of 'tax' and 'deduct' (common attributes between different search results a_i in A) that are highlighted in dark-grey, like 'overlook'.

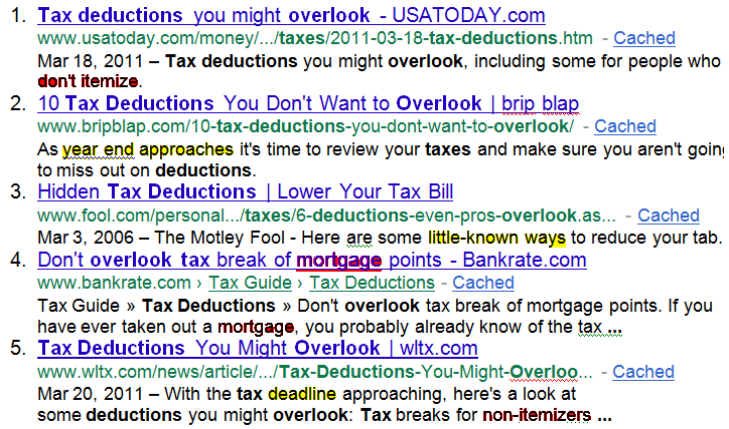
4) Extending the pair (tax-deduct) to a number of triples by adding, in particular, the newly acquired attribute "overlook": *Tax-deduct-overlook.*



1. TurboTax® - **Tax Deduction** Wisdom - Should You Itemize?
   turbotax.intuit.com › Tax Calculators & Tips - Cached
   Learn whether itemizing your **deductions** makes sense, or if you should simply take t
   no-questions-asked standard **deduction**. The standard **deduction** is ...
2. 10 big **deductions** too many of us miss - **tax** preparation - MSN Money
   money.msn.com/**taxes**/10-big-**deductions**-too-many-of-us-miss-schn... - Cached
   A lot of taxpayers don't know they can save thousands of dollars with these **tax** break
   Did you forget about any of these **deductions** and credits?
3. The Most-**Overlooked** **Tax Deductions**
   www.kiplinger.com/.../the-mostoverlooked-**tax-deductions**.html - Cached
   Every year, the IRS dutifully reports the most common blunders that taxpayers make
   their returns. And every year, at or near the top of the "oops" list ...
4. **Tax** Credits and **Deductions**
   taxes.about.com/od/**deductions**credits/**Deductions**_Credits.htm - Cached
   Lower your **tax** bill by taking advantage of **deductions** and **tax credits**. Tips for
   preparing your**taxes**.
5. **Tax** Topics - Topic 503 **Deductible** **Taxes**
   www.irs.gov/**tax**topics/tc503.html - Cached
   Feb 7, 2011 – To be **deductible**, the **tax** must be **imposed on you** and must have bee
   paid during your **tax** year. However, tables are available to determine ...
6. Tax - **Tax Deductions** - H&R Block
   www.hrblock.com/**taxes**/**tax**.../**deductions**.../**overlooked**_**deductions**... - Cached
   Find information on **tax deductions** from H&R Block.
7. What is a **Tax Deduction**?
   www.wisegeek.com/what-is-a-**tax-deduction**.htm - Cached
   May 13, 2011 – A **tax deduction reduces** the taxes a person must pay by a certain
   percentage. **Tax deductions** are different from tax credits, which...

Fig. 3a: First step of taxonomy learning, given the seed *tax-deduct*.

Fig 3b shows some answers for the extended path *tax-deduct- overlook.*



1. **Tax deductions** you might **overlook** - USATODAY.com
   www.usatoday.com/money/.../**taxes**/2011-03-18-**tax-deductions**.htm - Cached
   Mar 18, 2011 – **Tax deductions** you might **overlook**, including some for people who
   don't itemize.
2. 10 **Tax Deductions** You Don't Want to **Overlook** | brip blap
   www.bripblap.com/10-**tax-deductions**-you-dont-want-to-**overlook**/ - Cached
   As **year end approaches** it's time to review your **taxes** and make sure you aren't goin
   to miss out on **deductions**.
3. Hidden **Tax Deductions** | Lower Your Tax Bill
   www.fool.com/personal.../**taxes**/6-**deductions**-even-pros-**overlook**.as... - Cached
   Mar 3, 2006 – The Motley Fool - Here are some **little-known ways** to reduce your tab.
4. Don't **overlook tax** break of **mortgage** points - Bankrate.com
   www.bankrate.com › Tax Guide › Tax Deductions - Cached
   Tax Guide » **Tax Deductions** » Don't **overlook** tax break of mortgage points. If you
   have ever taken out a **mortgage**, you probably already know of the tax ...
5. **Tax Deductions** You Might **Overlook** | wltx.com
   www.wltx.com/news/article/.../**Tax-Deductions**-You-Might-**Overloo**... - Cached
   Mar 20, 2011 – With the **tax deadline** approaching, here's a look at
   some **deductions** you might **overlook**: **Tax** breaks for **non-itemizers** ...

Fig 3b. Search of extension of the taxonomy tree for the path *tax-deduct- overlook*.

The learning steps now are as follows:
1) Get search results for "*tax deduct overlook";*
2) Select candidate attributes (now, modifiers of terms from the current taxonomy path)
3) Turn candidate attributes into parameters by finding common expressions such as 'PRP-mortgage' in our case
4) Extend the taxonomy path by adding newly acquired parameters
Tax-deduct-overlook - ***mortgage,***
Tax-deduct- overlook - ***no_itemize.***

Having built the full taxonomy, we can now apply it to filter out search results, which are not *covered by* the taxonomy paths properly. Consider a query '*Can I deduct tax on mortgage escrow account?'* Fig. 3c shows the answers obtained. Two answers (shown in an oval frame) are irrelevant, because they do not include the taxonomy nodes {*deduct, tax , mortgage, escrow_account*}. Notice that the closest taxonomy path to the query is *tax – deduct – overlook-mortgage- escrow_account.*

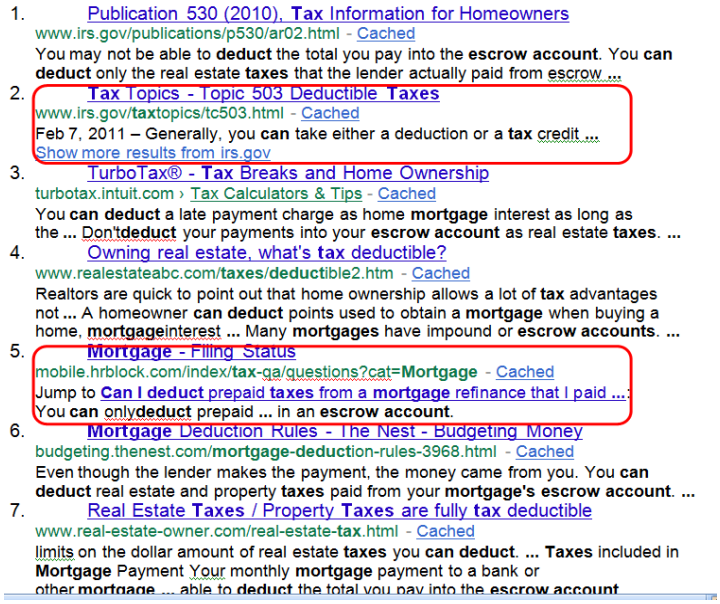

1. Publication 530 (2010), Tax Information for Homeowners
www.irs.gov/publications/p530/ar02.html - Cached
You may not be able to **deduct** the total you pay into the **escrow account**. You **can deduct** only the real estate **taxes** that the lender actually paid from escrow ...
2. Tax Topics - Topic 503 Deductible Taxes
www.irs.gov/**tax**topics/tc503.html - Cached
Feb 7, 2011 – Generally, you **can** take either a deduction or a **tax** credit ...
Show more results from irs.gov
3. TurboTax® - Tax Breaks and Home Ownership
turbotax.intuit.com › Tax Calculators & Tips - Cached
You **can deduct** a late payment charge as home **mortgage** interest as long as the ... Don't**deduct** your payments into your **escrow account** as real estate **taxes**. ...
4. Owning real estate, what's tax deductible?
www.realestateabc.com/**taxes**/**deduct**ible2.htm - Cached
Realtors are quick to point out that home ownership allows a lot of **tax** advantages not ... A homeowner **can deduct** points used to obtain a **mortgage** when buying a home, **mortgage**interest ... Many **mortgages** have impound or **escrow accounts**. ...
5. Mortgage - Filing Status
mobile.hrblock.com/index/**tax**-qa/questions?cat=**Mortgage** - Cached
Jump to **Can I deduct** prepaid **taxes** from a **mortgage** refinance that I paid ...
You **can** only**deduct** prepaid ... in an **escrow account**.
6. Mortgage Deduction Rules - The Nest - Budgeting Money
budgeting.thenest.com/**mortgage-deduct**ion-rules-3968.html - Cached
Even though the lender makes the payment, the money came from you. You **can deduct** real estate and property **taxes** paid from your **mortgage's escrow account**. ...
7. Real Estate Taxes / Property Taxes are fully tax deductible
www.real-estate-owner.com/real-estate-**tax**.html - Cached
limits on the dollar amount of real estate **taxes** you **can deduct**. ... **Taxes** included in **Mortgage** Payment Your monthly **mortgage** payment to a bank or other **mortgage** ... able to **deduct** the total you pay into the **escrow account**

Fig. 3c: Filtering out irrelevant Google answers using the built taxonomy

sell_hobby=>[[deductions, collection], [making, collection], [sales, business, collection], [collectibles, collection], [loss, hobby, collection], [item, collection], [selling, business, collection], [pay, collection], [stamp, collection], [deduction, collection], [car, collection], [sell, business, collection], [loss, collection]]

benefit=>[[office, child, parent], [credit, child, parent], [credits, child, parent], [support, child, parent], [making, child, parent], [income, child, parent], [resides, child, parent], [taxpayer, child, parent], [passed, child, parent], [claiming, child, parent], [exclusion, child, parent], [surviving, benefits, child, parent], [reporting, child, parent],

hardship=>[[apply, undue], [taxpayer, undue], [irs, undue], [help, undue], [deductions, undue], [credits, undue], [cause, undue], [means, required, undue], [court, undue].

Fig. 3d: Three sets of paths for the tax topic entities *sell hobby, benefit, hardship*

Fig. 3d shows the snapshot of taxonomy tree for three entities. For each entity, given the sequence of keywords, the reader can reconstruct the meaning in the context of tax domain. This snapshot illustrates the idea of taxonomy-based search relevance improvement: once the particular meaning (content, taxonomy path in our model) is established, we can find relevant answers. The head of the expression occurs in every path it yields (like {*sell_hobby – deductions - collection*}, {*sell_hobby – making – collection*}).

3.2 Generalization of sentences to extract keywords

Consider three sentences:

I am curious how to use the digital zoom of this camera for filming insects.
How can I get short focus zoom lens for digital camera?
Can I get auto focus lens for digital camera?

Fig. 4a shows the parse trees for these sentences. We generalize them by determining their maximal common sub-trees. Some sub-trees are shown as lists for brevity. The second and third trees are quite similar. Therefore, it is simple to build their common sub-tree as an (interrupted) path of the tree (Figs. 4a, 4b)

{ *MD-can, PRP-I, VB-get, NN-focus, NN-lens, IN-for JJ-digital NN-camera* }.

At the phrase level, we obtain:
`Noun phrases: [ [NN-focus NN-* ], [JJ-digital NN-camera ]]`
`Verb phrases: [ [VB-get NN-focus NN-* NN-lens IN-for JJ-digital NN-camera ]]`. Here the generalization of distinct values is denoted by '*'.

The common words remain in the maximum common sub-tree, except 'can', which is unique to the second sentence, and the modifiers of 'lens', which are different between the two sentences (shown as *NN-focus NN-* NN-lens*) . We generalize sentences that are less similar than sentences 2 and 3 on a phrase-by-phrase basis. Below, the syntactic parse tree is expressed via chunking (Abney 1991), using the format <position (POS – phrase)>.

Parse 1 `0(S-I am curious how to use the digital zoom of this camera for filming insects), 0(NP-I), 2(VP-am curious how to use the digital zoom of this camera for filming insects), … 2(VBP-am),`

Parse 2 `[0(SBARQ-How can I get short focus zoom lens for digital camera), 0(WHADVP-How), 0(WRB-How), 4(SQ-can I get short focus`

zoom lens for digital camera), 4(MD-can), 8(NP-I), 8(PRP-I),
10(VP-get short focus zoom lens for digital camera),…

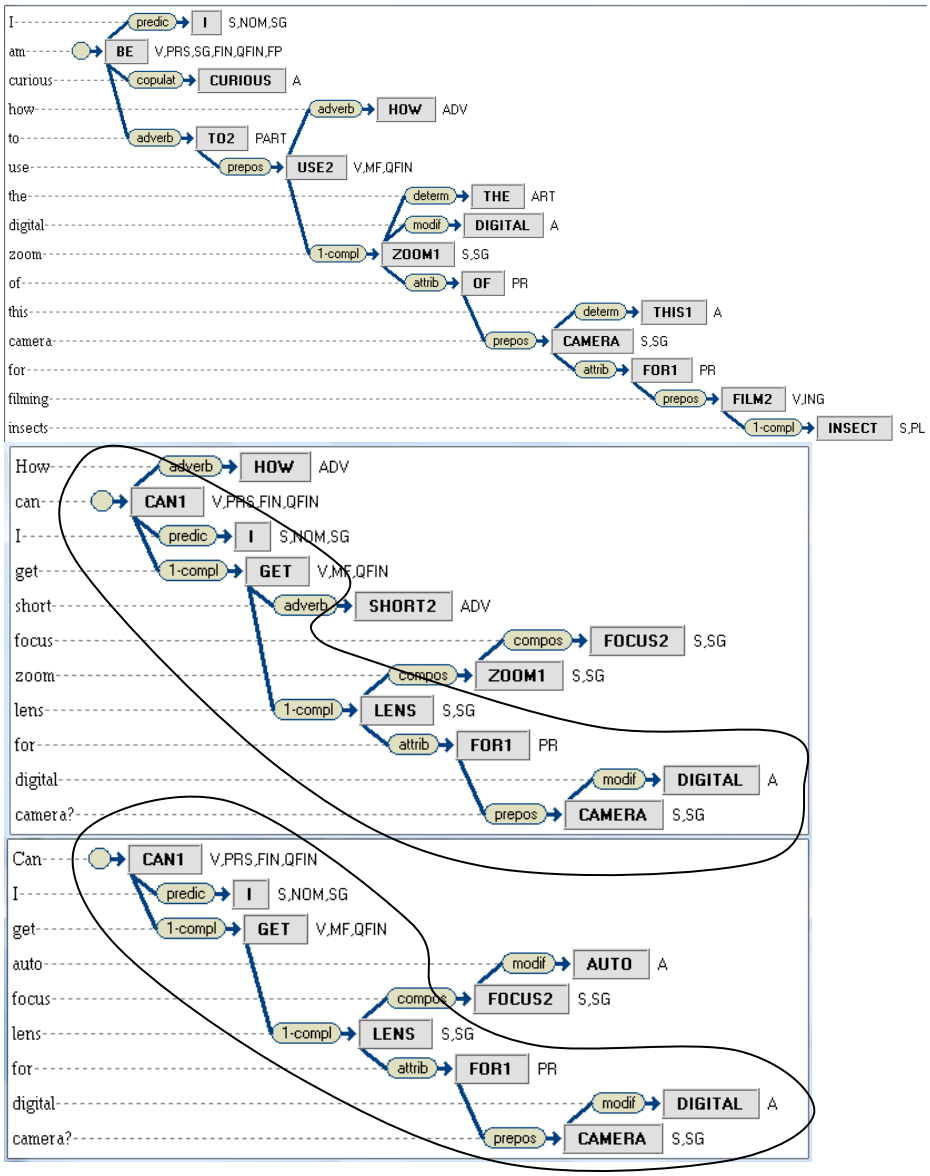


Fig. 4a: Parse trees for three sentences. The curve shows the common sub-tree (in this case, there is only one) for the second and third sentences.

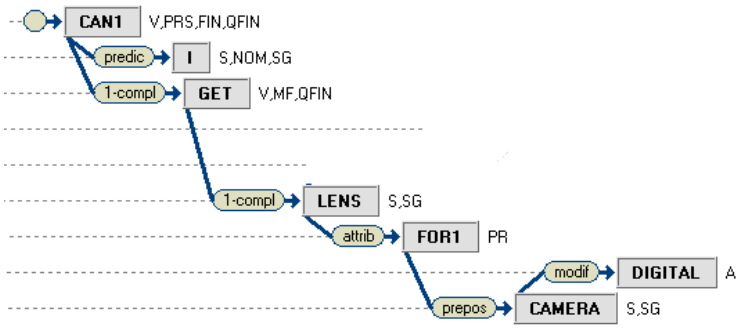


Fig. 4b: Generalization results for the second and third sentences

Next, we group the above phrases by their phrase type [NP, VP, PP, ADJP, WHADVP]. The numbers at the beginning of each phrase encode their character positions. Each group contains the phrases of the same type because the matches occur between the same types.

Grouped phrases 1

```
[[NP [DT-the JJ-digital NN-zoom IN-of DT-this NN-camera ], NP
[DT-the JJ-digital NN-zoom ], NP [DT-this NN-camera ], NP [VBG-
filming NNS-insects ]], [VP [VBP-am ADJP-curious WHADVP-how TO-
to VB-use DT-the JJ-digital NN-zoom IN-of DT-this NN-camera IN-
for VBG-filming NNS-insects ], VP [TO-to VB-use DT-the JJ-
digital NN-zoom IN-of DT-this NN-camera IN-for VBG-filming NNS-
insects ], VP [VB-use DT-the JJ-digital NN-zoom IN-of DT-this
NN-camera IN-for VBG-filming NNS-insects]
```

Grouped phrases 2

```
[[NP [JJ-short NN-focus NN-zoom NN-lens ], NP [JJ-digital NN-
camera ]], [VP [VB-get JJ-short NN-focus NN-zoom NN-lens IN-for
JJ-digital NN-camera ]], [], [PP [IN-for JJ-digital NN-camera
]],]
```

Generalization between phrases

The resultant generalization is shown in bold below for verb phrases (VP).

Generalization result

```
NP [ [JJ-* NN-zoom NN-* ],  [JJ-digital NN-camera ]]
VP [ [VBP-* ADJP-* NN-zoom NN-camera ],  [VB-* JJ-* NN-zoom
NN-* IN-for NN-* ]
PP [ [IN-* NN-camera ],  [IN-for NN-* ]]
```

Next we compute score for the generalizations:

$$score(NP) = (W_{<POS,*>} + W_{NN} + W_{<POS,*>}) + (W_{NN} + W_{NN}) = 3.4,$$

score(VP) = (2* $W_{<POS,*>}$ + 2*W_{NN})+ (4$W_{<POS,*>}$ +W_{NN} +W_{PRP}) = 4.55, and
score(PRP) = ($W_{<POS,*>}$+ W_{NN})+(W_{PRP}+W_{NN}) = 2.55,
therefore, score = 10.5.

Thus, such a common concept as '*digital camera*' is automatically generalized from the examples, as well as the verb phrase "*be some-kind-of zoom camera*". The latter generalization expresses a common meaning between this pair of sentences.

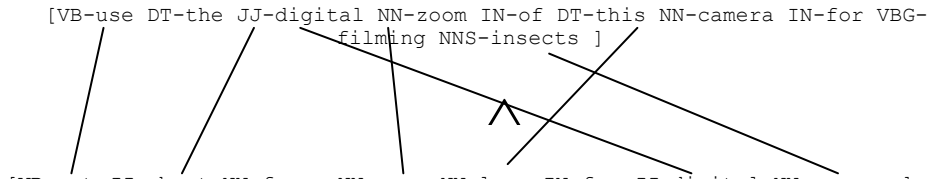
Note the occurrence of the expression [digital-camera] in the first sentence: although *digital* does not refer to *camera* directly, when we merge the two noun groups, *digital* becomes one of the adjectives of the resultant noun group with the head *camera.* It is matched against the noun phrase reformulated in a similar way (but with the preposition *for*) in the second sentence with the same head noun *camera.*

At the phrase level, generalization starts from setting correspondences between as many words as possible in the two phrases. Two phrases are aligned only if their head nouns are matched. A similar integrity constraint applies to aligning verb phrases, prepositional phrases and other types of phrases.

We now generalize two phrases, and denote the generalization operator as '$\wedge$'. Six mapping links between the phrases correspond to the six members of the generalization result phrase

[VB-* JJ-* NN-zoom NN-* IN-for NN-*].

Notice that only NN-zoom and IN-for remain as the same words, for the rest only part of speech information is retained.



=
[VB-* JJ-* NN-zoom NN-* IN-for NN-*]

3.3 Evaluation of search relevance improvement

3.3.1 Experimental setup

We evaluated relevance of taxonomy and syntactic generalization–enabled search engine based on Yahoo and Bing search engine APIs for the vertical domains of tax, investment, and retirement (Galitsky 2003).

For an individual query, the *relevance* was estimated as a percentage of correct hits among the first ten hits, using the values: {correct, marginally correct, incorrect} that is in line with the approach in (Resnik, Lin 2010). *Accuracy of a single search* session is calculated as the percentage of correct search results plus half of the percentage of marginally correct search results. *Accuracy of a particular search setting* (query type and search engine type) is calculated, averaging through 20 search sessions.

We also used customers' queries to eBay entertainment and product-related domains, from simple questions referring to a particular product, a particular user need, as well as a multi-sentence forum-style request to share a recommendation. The set of queries was split into noun-phrase class, verb-phrase class, how-to class, and also independently split in accordance to query length (from 3 keywords to multiple sentences). We ran 450 search sessions for evaluations of Sections 2.4 and 2.6.

To compare the relevance values between search settings, we used first 100 search results obtained for a query by Yahoo and Bing APIs, and then re-sorted them according to the score of the given search setting (*syntactic generalization* score and *taxonomy-based score*). To evaluate the performance of such a hybrid system, we used the *weighted sum of these two scores* (the weights were optimized in an earlier search sessions).

3.3.2 Evaluation of improvement of vertical search

Table 1 shows the results of evaluation of search relevance in the domain of vertical product search. One can see that taxonomy contributes significantly in relevance improvement, compared to domain-independent syntactic generalization. Relevance of the hybrid system that combines both of these techniques is improved by 14.8±1.1%.

The general conclusion is that for a vertical domain, a taxonomy should be definitely applied, and the syntactic generalization possibly applied, for improvement of relevance for all kinds of questions.

Notice from the Table 1 results that syntactic generalization usually improves the relevance on its own, and as a part of a hybrid system in a vertical domain where the taxonomy coverage is good (most questions are mapped well into taxonomy).

Taxonomy-based method is always helpful in a vertical domain, especially for a short queries (where most keywords are represented in the taxonomy) and multi-sentence queries (where the taxonomy helps to find the important keywords for matching with a question).

Query	phrase sub-type	Relevancy of baseline Yahoo search, %, averaging over 20 searches	Relevancy of baseline Bing search, %, averaging over 20 searches	Relevancy of re-sorting by generalization, %, averaging over 20 searches	Relevancy of re-sorting by using taxonomy, %, averaging over 20 searches	Relevancy of re-sorting by using taxonomy and generalization, %, averaging over 20 searches	Relevancy improvement for hybrid approach, comp. to baseline (averaged for Bing & Yahoo)
3-4 word phrases	noun phrase	86.7	85.4	87.1	93.5	93.6	1.088
	verb phrase	83.4	82.9	79.9	92.1	92.8	1.116
	how-to expression	76.7	78.2	79.5	93.4	93.3	1.205
	average	82.3	82.2	82.2	93.0	93.2	1.134
5-10 word phrases	noun phrase	84.1	84.9	87.3	91.7	92.1	1.090
	verb phrase	83.5	82.7	86.1	92.4	93.4	1.124
	how-to expression	82.0	82.9	82.1	88.9	91.6	1.111
	average	83.2	83.5	85.2	91.0	92.4	1.108
2-3 sentences	one verb one noun phrases	68.8	67.6	69.1	81.2	83.1	1.218
	both verb phrases	66.3	67.1	71.2	77.4	78.3	1.174
	one sent of how-to type	66.1	68.3	73.2	79.2	80.9	1.204
	average	67.1	67.7	71.2	79.3	80.8	1.199

Table 1: Improvement of accuracy in a vertical domain. The 3rd and 4th columns on the left are the baseline; the right-most column shows improvement of search accuracy.

3.3.3 Evaluation of horizontal web search relevance improvement

In a horizontal domain (searching for broad topics in finance-related and product-related domains of eBay) contribution of taxonomy is comparable to syntactic generalization (Table 2). Search relevance is improved by a 4.6±0.8% by a hybrid system, and is determined by a type of phrase and a query length.

Query	phrase sub-type	Relevancy of baseline Yahoo search, %, averaging over 20 searches	Relevancy of baseline Bing search, %, averaging over 20 searches	Relevancy of re-sorting by generalization, %, averaging over 20 searches	Relevancy of re-sorting by using taxonomy, %, averaging over 20 searches	Relevancy of re-sorting by using taxonomy and generalization, %, averaging over 20 searches	Relevancy improvement for hybrid approach, comp. to baseline (averaged for Bing & Yahoo)
3-4 word phrases	noun phrase	88.1	88.0	87.5	89.2	89.4	1.015
	verb phrase	83.4	82.9	79.9	80.5	84.2	1.013
	how-to expression	76.7	81.2	79.5	77.0	80.4	1.018
	average	82.7	84.0	82.3	82.2	84.7	1.015
5-6 word phrases	noun phrase	86.3	85.4	87.3	85.8	88.4	1.030
	verb phrase	84.4	85.2	86.1	88.3	88.7	1.046
	how-to expression	83.0	82.9	82.1	84.2	85.6	1.032
	average	84.6	84.5	85.2	86.1	87.6	1.036
7-8 word phrases	noun phrase	78.4	79.3	81.1	82.8	83.0	1.053
	verb phrase	75.2	73.8	74.3	78.3	79.2	1.063
	how-to expression	73.2	73.9	74.5	77.8	76.3	1.037
	average	75.6	75.7	76.6	79.6	79.5	1.051
8-10 word single sentences	noun phrase	68.8	67.9	71.2	69.7	72.4	1.059
	verb phrase	65.8	67.2	73.6	70.2	73.1	1.099
	how-to expression	64.3	63.9	65.7	67.5	68.1	1.062
	average	66.3	66.3	70.2	69.1	71.2	1.074
2 Sentences, >8 words total	1 verb 1 noun phrases	66.5	67.2	66.9	69.2	70.2	1.050
	both verb phrases	65.4	63.9	65.0	67.3	69.4	1.073
	one sent of how-to type	65.9	66.7	66.3	65.2	67.9	1.024
	average	65.9	65.9	66.1	67.2	69.2	1.049
3 sentences, >12 words total	1 verb 1 noun phrases	63.6	62.9	64.5	65.2	68.1	1.077
	both verb phrases	63.1	64.7	63.4	62.5	67.2	1.052
	one sent of how-to type	64.2	65.3	65.7	64.7	66.8	1.032
	average	63.6	64.3	64.5	64.1	67.4	1.053

Table 2: Evaluation of search relevance improvement in a horizontal domain

The highest relevance improvement is for longer queries and for multi-sentence queries. Noun phrases perform better at the baseline (Yahoo & Bing search engine APIs) and a hybrid system, than the verb phrases and the how-to phrases. Note that generalization can decrease relevance for short queries, where linguistic information is not as important as frequency analysis.

One can see from Table 2 that the hybrid system almost always outperforms the individual components. Thus, for a horizontal domain, syntactic generalization is a must and taxonomy is helpful for some queries, which happen to be covered by this taxonomy, and is useless for the majority of queries.

We observed that a taxonomy is beneficial for a queries in many forms, and their complexities. In contrast, syntactic generalization–supported search is beneficial for rather complex queries, exceeding 3-4 keywords. Taxonomy-based search is essential for a product search without requiring explicit use of their features and/or needs Conversely, syntactic generalization is sensitive to proper handling of phrasings in product names, matching the template:

product_name for *super-product* with *parts-of –product.*

We conclude that building taxonomy for such domain as product search is a plausible and rewarding task, and should be done for all kinds of product searches.

3.4 Multi-lingual taxonomy use

Syntactic generalization was deployed and evaluated in the framework of a Unique European Citizens' attention service (iSAC6+) project, an EU initiative to build a recommendation search engine in a vertical domain. As a part of this initiative, a taxonomy was built to improve the search relevance (easy4.udg.edu/isac/eng/index.php, de la Rosa, 2007). Taxonomy learning of the tax domain was conducted in English and then translated in Spanish, French, German and Italian. It was evaluated by project partners using the tool in Figs. 5a and 5b. To improve search precision a project partner in a particular location modifies the automatically learned taxonomy to fix a particular case, upload the taxonomy version adjusted for a particular location (Fig. 5a) and verify the improvement of relevance. An evaluator is able to sort search results by the original Yahoo score, the syntactic generalization score, and the taxonomy score to get a sense of how each of these scores work and how they correlate with the best order of answers for the best relevance (Fig 5b).

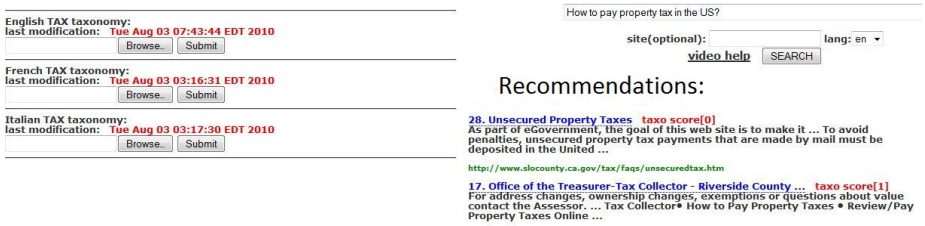
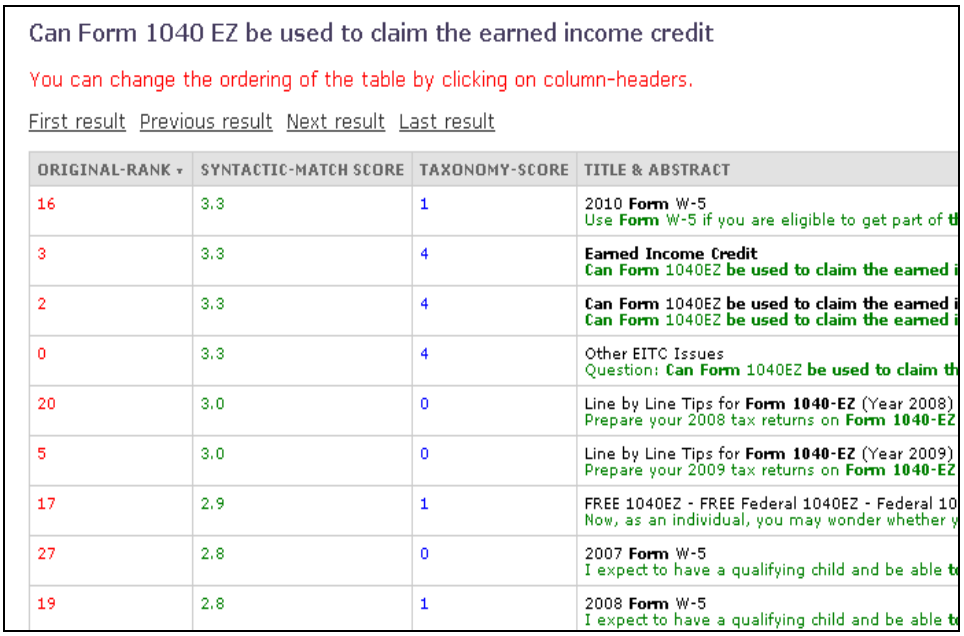


Fig. 5a: Tool for manual taxonomy adjustment for Citizens Recommendation Services



Can Form 1040 EZ be used to claim the earned income credit

You can change the ordering of the table by clicking on column-headers.

First result Previous result Next result Last result

ORIGINAL-RANK	SYNTACTIC-MATCH SCORE	TAXONOMY-SCORE	TITLE & ABSTRACT
16	3.3	1	2010 **Form** W-5 Use **Form** W-5 if you are eligible to get part of **t**
3	3.3	4	**Earned Income Credit** **Can Form** 1040EZ **be used to claim the earned i**
2	3.3	4	**Can Form** 1040EZ **be used to claim the earned i** **Can Form** 1040EZ **be used to claim the earned i**
0	3.3	4	Other EITC Issues Question: **Can Form** 1040EZ **be used to claim th**
20	3.0	0	Line by Line Tips for **Form 1040-EZ** (Year 2008) Prepare your 2008 tax returns on **Form 1040-EZ**
5	3.0	0	Line by Line Tips for **Form 1040-EZ** (Year 2009) Prepare your 2009 tax returns on **Form 1040-EZ**
17	2.9	1	FREE 1040EZ - FREE Federal 1040EZ - Federal 10 Now, as an individual, you may wonder whether y
27	2.8	0	2007 **Form** W-5 I expect to have a qualifying child and be able **t**
19	2.8	1	2008 **Form** W-5 I expect to have a qualifying child and be able **t**

Fig. 5b: Sorting search results by taxonomy-based and syntactic generalization scores for a given query "Can Form 1040 EZ be used to claim the earned income credit?"

3.5 Commercial evaluation of taxonomy-based text similarity

We subject the proposed technique of taxonomy-based and syntactic generalization-based techniques to commercial mainstream news analysis at AllVoices.com (Fig. 6a). The task is to *cluster* relevant news items together by means of finding similarity between the titles and first paragraphs, similarly to what we have done with questions and answers. By definition, multiple news articles belong to the same cluster if there is a substantial overlap in geographical locations, the names of individuals, organizations, other agents, and the relationships between them. Some of these can be extracted using entity taggers and/or taxonomies built offline, and some are handled in real time using syntactic generalization (the bottom of Fig. 6b). The latter is applicable if there is a lack of prior entity information.

In addition, syntactic generalization and taxonomy match was used to aggregate relevant images and videos from different sources, such as Google Image, YouTube and Flickr. It was implemented by assessing their relevance given their textual descriptions and tags. The precision of the text analysis is achieved by the site's usability (click rate): more than nine million unique visitors per month. If the precision were low, we assume that users would not click through to irrelevant 'similar' articles. Recall is accessed manually; however, the system needs to find at least a few articles, images and videos for each incoming article. Recall is generally not an issue for web mining

and web document analysis (it is assumed that there is a sufficiently high number of articles, images and videos on the web for mining).

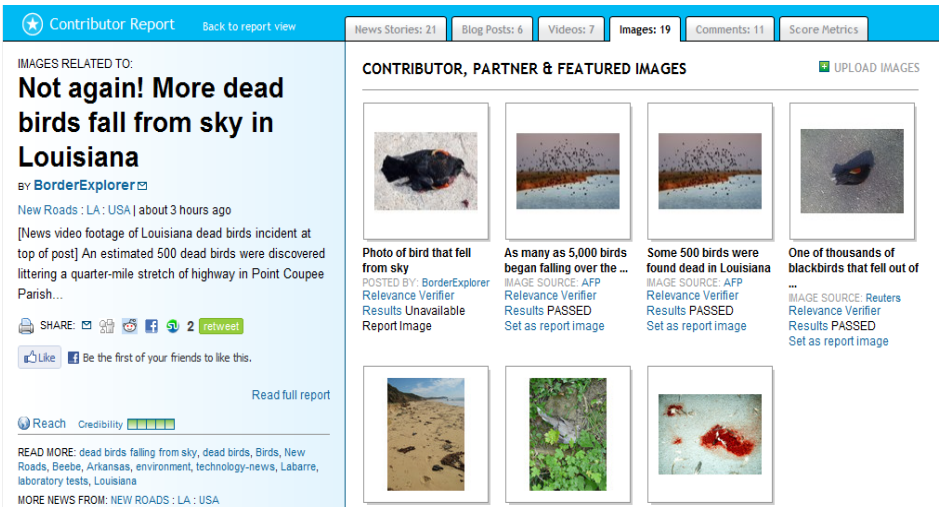


Fig. 6a: News articles and aggregated images found on the web and determined to be relevant to this article.

Relevance is ensured in two steps. First, we form a query to the image/video/blog search engine API, given an event title and first paragraph and extracting and filtering noun phrases by certain significance criteria. Second, we apply a similarity assessment to the texts returned from images/videos/blogs and ensure that substantial common noun, verb or prepositional sub-phrases can be identified between the seed events and these media (Fig. 6c).

The precision data for the relevance relationships between an article and other articles, blog postings, images and videos are presented in Table 3. Note that by itself, the taxonomy-based method has a very low precision and does not outperform the baseline of the statistical assessment. This baseline is based on TF*IDF model for keyword-based assessment of relevance. However, there is a noticeable improvement in the precision of the hybrid system, where the major contribution of syntactic generalization is improved by a few percentage points by the taxonomy-based method (Galitsky et al 2011). We can conclude that syntactic generalization and the taxonomy-based methods (which also rely on syntactic generalization) use different sources of relevance information. Therefore, they are complementary to each other.

The objective of syntactic generalization is to filter out false-positive relevance decisions made by a statistical relevance engines. This statistical engine has been designed following (Liu & Birnbaum 2007, Liu & Birnbaum 2008). The percentage of false-positive news stories was reduced from 29% to 17% (approximately 30000 stories/month, viewed by 9 million unique users), and the percentage of false positive image attachment was reduced from 24% to 20% (approximately 3000 images and 500 videos attached to stories monthly). The percentages shown are (100% - precision

values); recall values are not as important for web mining, assuming there is an unlimited number of resources on the web and that we must identify the relevant ones.

Gay Pulitzer Prize-Winning Reporter Jose Antonio Vargas Comes Out as ...

Towleroad | about 10 hours ago

Gay Pulitzer Prize-Winning Reporter Jose Antonio Vargas Comes Out as Undocumented Immigrant Jose Antonio Vargas, a gay journalist who won a Pulitzer Prize for his coverage of the Virginia Tech shootings in the Washington Post

Fig. 6b Syntactic generalization result for the seed articles and the other article mined for on the web.

Our approach belongs to the category of structural machine learning. The accuracy of our approach is worth comparing with the other parse tree learning approach based on the statistical learning of SVM. For instance, (Moschitti 2006) compares the performances of the bag-of-words kernel, syntactic parse trees and predicate argument structures kernel, and the semantic role kernel, confirming that the accuracy improves in this order and reaches an F-measure of 68% on the TREC dataset. Achieving comparable accuracies, the kernel-based approach requires manual adjustment. However, it does not provide similarity data in the explicit form of common sub-phrases. Structural machine learning methods are better suited for performance-critical production environments serving hundreds millions of users because they better fit modern software quality assurance methodologies. Logs of the discovered commonality expressions are

maintained and tracked, which ensures the required performance as the system evolves over time and the text classification domains change.

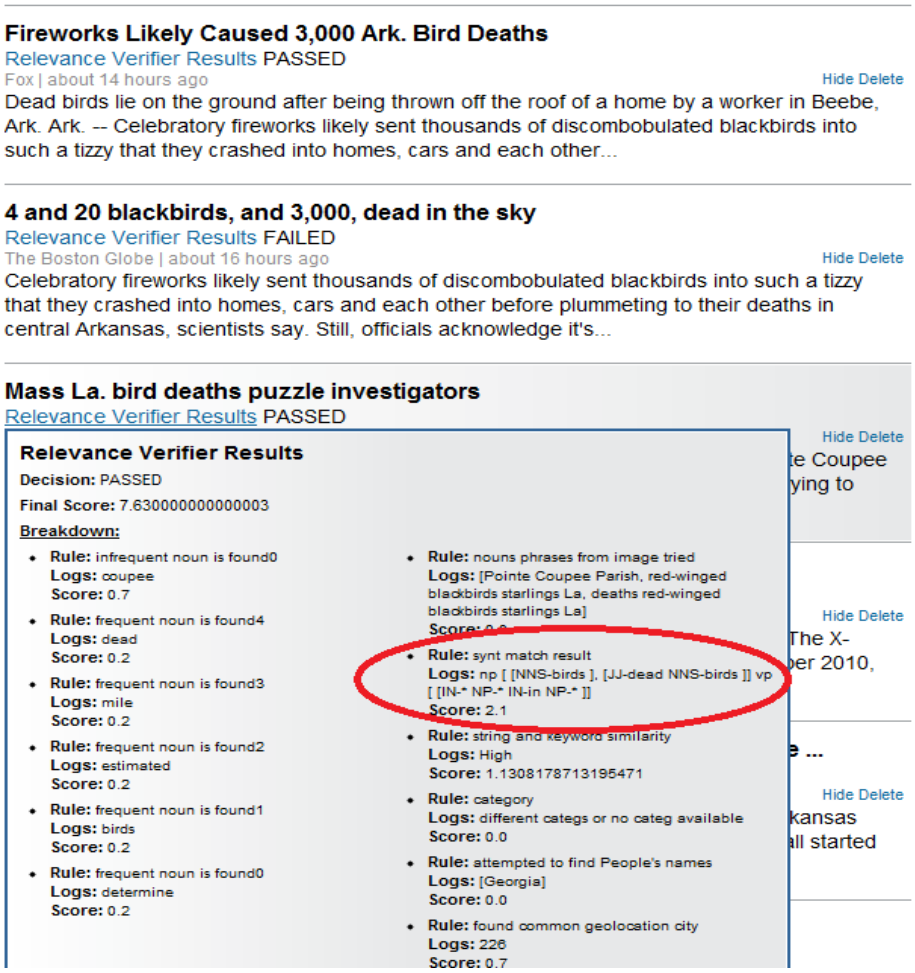


Fig. 6c: Explanation for relevance decision while forming a cluster of news articles for the one on Fig. 6a. The circled area shows the syntactic generalization result for the seed articles and the given one.

3.6 Opinion-oriented open search engine

The search engine based on syntactic generalization is designed to provide opinion data in an aggregated form obtained from various sources. This search engine uses conventional search results and Google-sponsored link formats that are already accepted by a vast community of users.

Media/ method of text similarity assessment	Full size news articles	Abstracts of articles	Blog posting	Comments	Images	Videos
Frequencies of terms in documents (baseline)	29.3%	26.1%	31.4%	32.0%	24.1%	25.2%
Syntactic generalization	19.7%	18.4%	20.8%	27.1%	20.1%	19.0%
Taxonomy-based	45.0%	41.7%	44.9%	52.3%	44.8%	43.1%
Hybrid Syntactic generalization and Taxonomy-based	17.2%	16.6%	17.5%	24.1%	20.2%	18.0%

Table 3: Improvement in the precision of text similarity

The user interface is shown in Fig. 7. To search for an opinion, a user specifies a product class, a name of particular products, and a set of its features, specific concerns, needs or interests. A search can be narrowed down to a particular source; otherwise, multiple sources of opinion (review portals, vendor-owned reviews, forums and blogs available for indexing) are combined.

The opinion search results are shown on the bottom left. For each result, a snapshot is generated indicating a product, its features that the system attempts to match to a user opinion request, and sentiments. In case of multiple sentence queries, a hit contains a combined snapshot of multiple opinions from multiple sources, dynamically linked to match the user request.

Automatically generated product ads compliant with the Google-sponsored link format are shown on the right. The phrases in the generated ads are extracted from the original products' web pages and may be modified for compatibility, compactness and their appeal to potential users. There is a one-to-one correspondence between the products in the opinion hits on the left and the generated ads on the right (unlike in Google, where the sponsored links list different websites from those presented on the left).

Both respective business representatives and product users are encouraged to edit and add ads, expressing product feature highlights and usability opinions, respectively. This feature assures openness and community participation in providing access to linked opinions for other users. A search phrase may combine multiple sentences: for example: "*I am a beginning user of digital cameras. I want to take pictures of my kids and pets. Sometimes I take it outdoors, so it should be waterproof to resist the rain*".

Fig. 7: User interface of the search engine used to evaluate taxonomies

Obviously, this type of specific opinion request can hardly be represented by keywords like ‘beginner digital camera kids pets waterproof rain’.

For a multi-sentence query, the results are provided as linked search hits:

Take Pictures of Your **Kids**? ... Canon 400D EOS Rebel XTI **digital** SLR **camera** review ↔ I am by no means a professional or long-time user of SLR cameras.

How To **Take Pictures** Of **Pets** And **Kids** … Need help with **Digital** slr **camera** please!!!? - Yahoo! Answers ↔ I am a **beginner** in the world of the **digital** SLR ...

Canon 400D EOS Rebel XTI **digital** SLR **camera** review (Website Design Tips) / Animal, **pet, children**, equine, livestock, farm portrait and stock ↔ I am a **beginner** to the slr **camera** world. ↔ I want to **take** the best **picture** possible because I know you.

Linking (**↔**) is determined in real time to address each part of a multi-sentence query, which may be a blog posting seeking advice. Linked search results provide comprehensive opinions on the topic of the user's interest, obtained from various sources and linked on the fly.

The problem of matching user needs while product search has been addressed in (Blanco-Fernández et al 2011, Galitsky et al 2009). An example of such user need expression would be 'a cell phone which fits in my palm'.

This need depends on the item's nature, the user's preferences, and time. Blanco-Fernández et al (2011) present a filtering strategy that exploits the semantics formalized in an ontology in order to link items (and their features) to time functions; the shapes of these functions are corrected by temporal curves built from the consumption stereotypes which are personalized to users.

4 Related work

4.1. Transfer learning paradigm for vertical and horizontal domains

In this paper we approached taxonomy building from the standpoint *of* **transfer learning paradigm** (Raina et al. 2006, Pan and Yang 2010). Although we build our taxonomy to function in a vertical domain, we use a horizontal domain for web mining to build it. For building taxonomies, transfer learning allows knowledge to be extracted from a wide spectrum of web domains and be used to enhance taxonomy-based search in a target domain. We will call these web domains auxiliary domains.

For transfer learning we compute the similarity between phrases in auxiliary and target domains using syntactic generalization as an extension of bag-of-words approach. The paper introduced a novel way for finding the structural similarity between sentences, to enable transfer learning at a structured knowledge level. This allows learning a non-trivial structural (semantic) similarity mapping between phrases in two different domains when they are completely different in terms of their vocabularies.

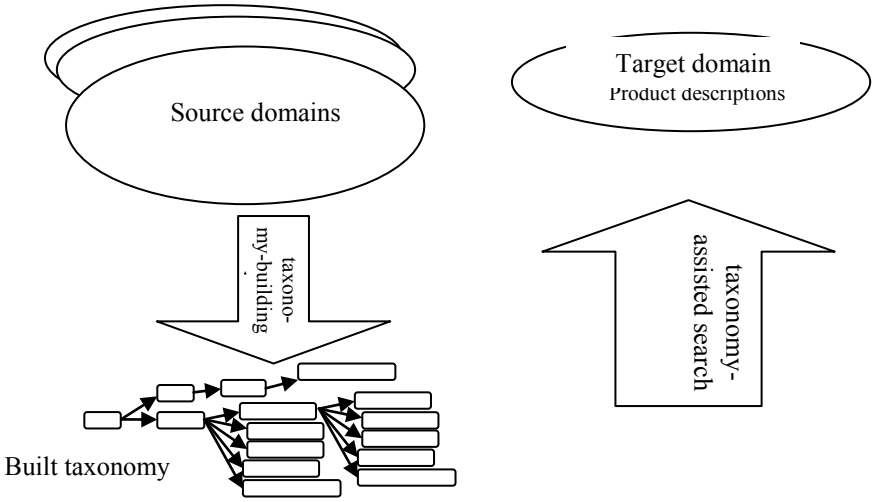


Fig.8: Taxonomy-assisted search viewed from the standpoint of transfer learning

It is usually insufficient to web mine documents for building taxonomies for vertical domains in this vertical domain only. Moreover, when a target domain includes social network data, micro-text, it is usually hard to find enough such data for building taxonomies within this domain, so a transfer learning methodology is required, which mines a wider set of domains with similar vocabulary. The transfer learning is then needs to be supported my matching syntactic expressions from distinct domains. In this study we perform it on the level of constituency parse trees (Fig. 8).

4.2. Taxonomies

WordNet is the most commonly used computational lexicon of English for word sense disambiguation, a task aimed to assigning the most appropriate senses (i.e. synsets) to words in context (Navigli 2009). It has been argued that WordNet encodes sense distinctions that are too fine-grained even for humans. This issue prevents WSD systems from achieving high performance. The granularity issue has been tackled by proposing clustering methods that automatically group together similar senses of the same word. Though WordNet contains a sufficiently wide range of common words, it does not cover special domain vocabulary. Since it is primarily designed to act as an underlying database for different applications, those applications cannot be used in specific domains that are not covered by WordNet.

A number of currently available general-purpose resources such as DBPEdia, Freebase, Yago assist entity-related searches, but are insufficient to filter out irrelevant answers concerning certain activity with an entity and its multiple parameters. A set of vertical ontologies, such as last.fm for artists, are also helpful for entity-based searches

in vertical domains, however their taxonomy trees are rather shallow, and usability for recognizing irrelevant answers is limited.

As text documents are massively available on the web as well as an access to them via **web search engine APIs**, most researchers have attempted to learn taxonomies on the basis of textual input. Several researchers explored taxonomic relations explicitly expressed in texts by pattern matching (Hearst 1992, Poesio et al 2002). One drawback of pattern matching is that it involves the predefined choice of semantic relations to be extracted.

In this study, to improve the flexibility of pattern matching we used transfer learning based on parse patterns, which is higher level of abstraction than sequences of words. We extend the notion of syntactic contexts from a partial cases such as noun + modifier and dependency triple (Lin 1998) towards finding a parse sub-tree in a parse tree. Our approach also extends handling of internal structure of noun phrases used to find taxonomic relations (Buitelaar 2003). Many researchers follow Harris' distributional hypothesis of correlation between semantic similarity of words or terms, and the extent to which they share similar syntactic contexts (Harris 1968). Clustering only requires a minimal amount of manual semantic annotation by a knowledge engineer, so clustering is frequently combined with pattern matching to be applied to syntactic contexts in order to also extract previously unexpected relations. We improve learning taxonomy on the web by combining supervised learning of the seed with unsupervised learning of the consecutive sets of relationships, also addressing such requirements of a taxonomy building process as evolvability and adaptability to new query domains of search engine users.

The current challenge in the area of taxonomy-supported searches is how to apply an imperfect taxonomy, automatically compiled from the web, to improve search. Lightweight keyword based approaches cannot address this challenge. This paper addresses it by using web mining to get training data for learning, and syntactic generalization as a learning tool.

This paper presented an **automated taxonomy building mechanism** which is based on initial set of main entities (a seed) for given vertical knowledge domain. This seed is then automatically extended by mining of web documents which include a meaning of a current taxonomy node. This node is further extended by entities which are the results of inductive learning of commonalities between these documents. These commonalities are extracted using an operation of syntactic generalization, which finds the common parts of syntactic parse trees of a set of documents, obtained for the current taxonomy node. Syntactic generalization has been extensively evaluated commercially to improve text relevance (Galitsky et al 2010, Galitsky et al 2011), and in this study we also apply it in the transfer learning setting for automated building of taxonomies.

Proceeding from **parsing to semantic level** is an important task towards natural language understanding, and has immediate applications in tasks such as information extraction and question answering (Allen 1987, Ravichandran and Hovy 2002, Dzikovska 2005, Wang et al 2009). In the last ten years there has been a dramatic shift in computational linguistics from manually constructing grammars and knowledge bases to partially or totally automating this process by using statistical learning methods trained on large annotated or non-annotated natural language corpora. However,

instead of using such corpora, in this paper we use web search results for common queries, since their accuracy is higher and they are more up-to-date than academic linguistic resources in terms of specific domain knowledge, such as tax.

The value of semantically-enabling search engines for improving search relevance has been well understood by the commercial search engine community (Heddon 2008). Once an 'ideal' taxonomy is available, properly covering all important entities in a vertical domain, it can be directly applied to filtering out irrelevant answers.

4.3. Comparative analysis of taxonomy-based systems

Table 4 presents the comparative analysis of some of taxonomy-based systems. It includes description of the taxonomy type, building mode, and its properties with respect to support of search, including filtering irrelevant answers by matching them with the query and taxonomy. In this table the current approach is the only one directly targeting filtering out irrelevant answers obtained by other components of a search engines.

5. Conclusion

We conclude that full-scale syntactic processing approach based on keyword taxonomy learning, and iterative taxonomy extension, is a viable way to enhance web search engines. Java-based OpenNLP component serves as an illustration of the proposed algorithm, and it is ready to be integrated with existing search engines.

In the future studies we plan to proceed from generalization of individual sentences to the level of paragraphs, deploying discourse theories and deeper analyzing the structure of text.

Class of taxonomy	How it is built	Supports search?	Support of reasoning and linguistic features?	How appropriate for filtering out irrelevant
WordNet	Manually groups English words into sets of synonyms called synsets, provides short, general definitions, and records the various semantic relations between these synonym sets	Query expansion	No reasoning but rich linguistic features	Not very
Learn ontological attributes from the Web (Sanchez 2010)	Auto, non-supervised and domain-independent methodology to extend ontological classes in terms of learning concept attributes, data-types, value ranges and measurement units.	Query expansion and suggested values set	Yes	To some degree
Multi-agent for Taxonomy-Based Web Search (Kerschberg et al 2003)	Users create personalized search taxonomies via the Weighted Semantic-Taxonomy Tree. Consulting a Web taxonomy agent such as WordNet helps refine the terms in the tree. The concepts represented in the tree are then transformed into a collection of queries processed by existing search engines.	Multi-Attribute-Based Search Preference	Yes, the concepts represented in the tree are then transformed into a collection of queries processed by existing search engines	To some degree
Content and Structure of a Term Taxonomy (Kozareva et al 2009)	Weakly supervised bootstraping algorithm that reads Web texts and learns taxonomy terms. The bootstrapping algorithm starts with two seed words (a seed hypernym (Root concept) and a seed hyponym) that are inserted into a doubly anchored hyponym pattern	No	Reasoning but not linguistic features	Not very
Concept learning-based taxonomies (Roth 2006)	Categorizing epistemic communities automatically and hierarchically, rebuilding a relevant taxonomy in the form of a hypergraph of significant epistemic sub-communities.	No	Reasoning but not linguistic features	Not very

General-purpose resources such as DBPEdia, Freebase, Yago	Contains relation between entities collected by a number of ways. Relations include similarity, tags, super- and subentity.	To some degree	Basic reasoning, no linguistic features	Not very
Textual-inference based support of search (Schoenmackers 2008)	Probabilistic approximate textual inference over tuples extracted from text. Utilizes sizable chunks of the Web corpus as source text. Taxonomy is constructed as a Markov network. The input a conjunctive query, a set of inference rules expressed as Horn clauses, and large sets of ground assertions extracted from the Web, WordNet, and other knowledge bases	no real time syntactic match is conducted	utilizes logical inference to find the subset of ground assertions and inference rules that may influence the answers to the query — enabling the construction of a focused Markov network.	Finds correct answers on its own, but does not filter incorrect ones
This work	Search-oriented taxonomies are built via web mining by employing machine learning of parse trees. Semi-supervised learning setting in a vertical domains, using search engine APIs.	Facilitates match between query and candidate answer	Features in parse tree and limited reasoning	Specifically designed for this purpose

Table 4: Comparative analysis of taxonomy-based systems with respect to support of search.

References

1. Alani, H. and Brewster, C. Ontology ranking based on the analysis of concept structures. K-CAP '05 Proceedings of the 3rd international conference on Knowledge Capture 2005.
2. R. Raina, A. Battle, H. Lee, B. Packer, and A.Y. Ng, "Self-Taught Learning: Transfer Learning from Unlabeled Data," Proc. 24th Int'l Conf. Machine Learning, pp. 759-766, June 2007.
3. Allen, J.F. Natural Language Understanding, Benjamin Cummings, 1987.
4. Antonio Moreno, Aida Valls, David Isern, Lucas Marin, Joan Borràs. Sig-Tur/E-Destination: Ontology-based personalized recommendation of Tourism and Leisure Activities.Engineering Applications of Artificial Intelligence. Available online17 March 2012.
5. Blanco-Fernández, Y., Martín López-Nores, José J. Pazos-Arias, Jorge García-Duque. An improvement for semantics-based recommender systems grounded on attaching temporal information to ontologies and user profiles. Engineering Applications of Artificial Intelligence. Volume 24, Issue 8, December 2011, Page 1385–1397.
6. Bong-Horng Chu, Cheng-En Lee, Cheng-Seen Ho. An ontology-supported database refurbishing technique and its application in mining actionable troubleshooting rules from real-life databases. Engineering Applications of Artificial Intelligence. Volume 21, Issue 8, December 2008, Pages 1430–1442.
7. Boris A. Galitsky, Boris Kovalerchuk, Josep Lluis de la Rosa. Assessing plausibility of explanation and meta-explanation in inter-human conflicts. A Special Issue on Semantic-based Information and Engineering Systems. Engineering Applications of Artificial Intelligence. Volume 24, Issue 8, December 2011, Pages 1472–1486.
8. Buitelaar P., D. Olejnik, and M. Sintek. A proteg´e´ plug-in for ontology extraction from text based on linguistic analysis. In Proceedings of the International Semantic Web Conference (ISWC), 2003.
9. Bunke, H. Graph-Based Tools for Data Mining and Machine Learning. Lecture Notes in Computer Science, 2003, Volume 2734/2003, 7-19.
10. Cardie, C., Mooney R.J. Machine Learning and Natural Language. Machine Learning 1(5) 1999.
11. Carlos Vicient, David Sánchez , Antonio Moreno. An automatic approach for ontology-based feature extraction from heterogeneous textual resources. Engineering Applications of Artificial Intelligence. Available online 12 September 2012.
12. Carreras X. and Luis Marquez. 2004. Introduction to the CoNLL-2004 shared task: Semantic role labeling. In Proceedings of the Eighth Conference on Computational Natural Language Learning, pp 89–97, Boston, MA. ACL.
13. Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze, Introduction to Information Retrieval, Cambridge University Press. 2008.

14. Cimiano, P., Aleksander Pivk, Lars Schmidt-Thieme, Steffen Staab. Learning Taxonomic Relations from Heterogeneous Sources of Evidence. In Buitelaar , P.,Cimiano, P. and Magnini, B., Eds.) Ontology Learning from Text: Methods, Evaluation and Applications. IOS Press 2004
15. David Sánchez, Antonio Moreno: Pattern-based automatic taxonomy learning from the Web. AI Commun. 21(1): 27-48 (2008).
16. David Sánchez: A methodology to learn ontological attributes from the Web. Data Knowl. Eng. 69(6): 573-597 (2010).
17. De la Rosa, JL, Mercè Rovira, Martin Beer, Miquel Montaner, and Denisa Gibovic, "Reducing Administrative Burden by Online Information and Referral Services", Citizens and E-Government: Evaluating Policy and Management, pp: 131-157, IGI Global, Editor: Christopher G. Reddick Austin, Texas, 2010.
18. Durme, B. V.; Huang, Y.; Kupsc, A.; and Nyberg, E. 2003. Towards light semantic processing for question answering. HLT Workshop on Text Meaning.
19. Dzikovska, M., M. Swift, Allen, J., de Beaumont, W. (2005). Generic parsing for multi-domain semantic interpretation. International Workshop on Parsing Technologies (Iwpt05), Vancouver BC.
20. Galitsky, B. Dobrocsi, G., de la Rosa, J.L. and Kuznetsov SO: Using Generalization of Syntactic Parse Trees for Taxonomy Capture on the Web. ICCS 2011: 104-117.
21. Galitsky, B. Dobrocsi, G., de la Rosa, J.L. Inferring semantic properties of sentences mining syntactic parse trees. Data & Knowledge Engineering, Volume 81-82, 21-45, 2012.
22. Galitsky, B. Disambiguation Via Default Rules Under Answering Complex Questions. Intl J AI Tools 14(1-2), World Scientific, 2005.
23. Galitsky, B. Natural Language Question Answering System: Technique of Semantic Headers. Advanced Knowledge International, Australia 2003.
24. Galitsky, B., Dobrocsi, G., de la Rosa, JL, Kuznetsov SO: From Generalization of Syntactic Parse Trees to Conceptual Graphs. ICCS 2010: 185-190.
25. Boris Galitsky: Machine learning of syntactic parse trees for search and classification of text. Eng. Appl. of AI 26(3): 1072-1091 (2013)
26. Grefenstette, G. Explorations in Automatic Thesaurus Discovery, Kluwer Academic 1994.
27. Harris, Z.. Mathematical Structures of Language. Wiley, 1968
28. Hearst, MA . Automatic acquisition of hyponyms from large text corpora. In Proceedings ofthe 14th International Conference on Computational Linguistics, 539–545, 1992.
29. Heddon, H. 2008 Better Living Through Taxonomies. Digital Web Magazine, www.digital-web.com/articles/better_living_through_taxonomies/
30. Howard, R.W. Classifying types of concept and conceptual structure: Some taxonomies. Journal of Cognitive Psychology, V4, Issue 2 April 1992, 81 - 111.
31. Kai Wang, Zhaoyan Ming, and Tat-Seng Chua. 2009. A syntactic tree matching approach to finding similar questions in community-based QA services. In Proceedings of the 32nd international ACM SIGIR conference on Research

and development in information retrieval (SIGIR '09). ACM, New York, NY, USA, 187-194.
32. Kapoor, S and H. Ramesh, Algorithms for Enumerating All Spanning Trees of Undirected and Weighted Graphs, SIAM J. Computing, vol. 24, pp. 247-265, 1995.
33. Kerschberg, L.,W. Kim, and A. Scime, "A Semantic Taxonomy-Based Personalizable Meta-Search Agent," in Innovative Concepts for Agent-Based Systems, vol. LNAI 2564, Lecture Notes in Artificial Intelligence, W. Truszkowski, Ed. Heidelberg: Springer-Verlag, 2003, pp. 3-31.
34. Kingsbury, P. and Palmer, M.: From Treebank to PropBank. In Proc. of the 3rd LREC, Las Palmas (2002).
35. Kozareva, Z., Eduard Hovy, and Ellen Riloff , Learning and Evaluating the Content and Structure of a Term Taxonomy. Learning by Reading and Learning to Read AAAI Spring Symposium 2009. Stanford CA.
36. Lin , D. Automatic retrieval and clustering of similar words, in Proc of COLING-ACL98, 1998.
37. Lin, D., and Pantel, P. 2001. DIRT: discovery of inference rules from text. In Proc. of ACM SIGKDD Conference on Knowledge Discovery and Data Mining 2001, 323–328.
38. Liu, J., Birnbaum, L.: What do they think?: Aggregating local views about news events and topics. WWW 2008: 1021-1022.
39. Liu, J., Birnbaum, L: Measuring Semantic Similarity between Named Entities by Searching the Web Directory. Web Intelligence 2007: 461-465.
40. López Arjona, AM, Miquel Montaner Rigall, Josep Lluís de la Rosa i Esteva, Maria Mercè Rovira i Regàs, POP2.0: A search engine for public information services in local government, ISSN: 0922-6389, Artificial Intelligence Research and Development, Vol:163 pp: 255-262, IOS Press, Amsterdam, 2007, Cecilio Angulo, Lluís Godo (eds).
41. Moschitti A., Daniele Pighin and Roberto Basili. Semantic Role Labeling via Tree Kernel joint inference. In Proceedings of the 10th Conference on Computational Natural Language Learning, New York, USA, 2006
42. Moschitti, A.Syntactic and Semantic Kernels for Short Text Pair Categorization. Proceedings of the 12th Conference of the European Chapter of the ACL. 2009.
43. Moschitti,A. Efficient Convolution Kernels for Dependency and Constituent Syntactic Trees. In Proceedings of the 17th European Conference on Machine Learning, Berlin, Germany, 2006.
44. OpenNLP opennlp.apache.org(2012)
45. Plotkin. GD A note on inductive generalization. In Meltzer and Michie, editors. Machine Intelligence, volume 5. Edinburgh University Press, 1970, pp 153–163
46. Poesio M., T. Ishikawa, S. Schulte im Walde, and R. Viera. Acquiring lexical knowledge for anaphora resolution. In Proceedings of the 3rd Conference on Language Resources and Evaluation (LREC), 2002.
47. R. Navigli. Word Sense Disambiguation: A Survey, ACM Computing Surveys, 41(2), 2009, pp. 1–69.

48. Ravichandran, D and Hovy E. 2002. Learning surface text patterns for a Question Answering system. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002), Philadelphia, PA
49. Reinberger, Marie-Laure, Peter Spyns: Generating and Evaluating Triples for Modelling a Virtual Environment. OTM Workshops 2005: 1205-1214.
50. Resnik, P. and Lin, J. (2010) Evaluation of NLP Systems, in The Handbook of Computational Linguistics and Natural Language Processing (eds A. Clark, C. Fox and S. Lappin), Wiley-Blackwell, Oxford, UK.
51. Roth, C. 2006 Compact, evolving community taxonomies using concept lattices ICCS 14 — July 17-21, 2006; Aalborg, DK.
52. Sinno Jialin Pan and Qiang Yang, A Survey on Transfer Learning, IEEE Transactions of Knowledge and Data Engineering, VOL. 22, NO. 10, OCTOBER 2010.
53. Stefan Schoenmackers, Oren Etzioni, and Daniel S. Weld.Scaling Textual Inference to the Web, EMNLP-2008.
54. Wu, Sheng-Tang, Li, Yuefeng, Xu, Yue, Pham, Binh and Chen, Yi-Ping Phoebe. Automatic pattern-taxonomy extraction for web mining, in IEEE/WIC International Conference on Web Intelligence (WI 2004) : Beijing, China, September 20-24, 2004
55. Xiang-Yang Wang, , Bei-Bei Zhang, Hong-Ying Yang. Active SVM-based relevance feedback using multiple classifiers ensemble and features reweighting. Engineering Applications of Artificial Intelligence. Available online 25 June 2012.
56. Zeng, Yi, Ning Zhong, Yan Wang, Yulin Qin, Zhisheng Huang, Haiyan Zhou, Yiyu Yao and Frank van Harmelen. User-centric query refinement and processing using granularity-based strategies. Knowledge and Information Systems V27, N 3, 419-450 (2010).