Engineering Applications of Artificial Intelligence ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at SciVerse ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Machine learning of syntactic parse trees for search and classification of text

Boris Galitsky *

eBay Inc. 2145, Hamilton Avenue, San Jose CA, USA

ARTICLE INFO

Article history:
Received 4 February 2012
Received in revised form
12 September 2012
Accepted 18 September 2012

Keywords:
Machine learning
Parse trees
Text classification
Text search

ABSTRACT

We build an open-source toolkit which implements deterministic learning to support search and text classification tasks. We extend the mechanism of logical generalization towards syntactic parse trees and attempt to detect weak semantic signals from them. Generalization of syntactic parse tree as a syntactic similarity measure is defined as the set of maximum common sub-trees and performed at a level of paragraphs, sentences, phrases and individual words. We analyze semantic features of such similarity measure and compare it with semantics of traditional anti-unification of terms. Nearest-neighbor machine learning is then applied to relate a sentence to a semantic class. Using syntactic parse tree-based similarity measure instead of bag-of-words and keyword frequency approach, we expect to detect a weak semantic signal otherwise unobservable. The proposed approach is evaluated in a four distinct domains where a lack of semantic information makes classification of sentences rather difficult. We describe a toolkit which is a part of Apache Software Foun-dation project OpenNLP, designed to aid search engineers in tasks requiring text relevance assessment.

1. Introduction

Ascending from the syntactic to semantic level is an important component of natural language (NL) understanding, and has immediate applications in tasks such information extraction and question answering (Allen, 1987; Cardie and Mooney, 1999; Ravichandran and Hovy, 2002. A number of studies demonstrated that the increase in the complexity of information retrieval (IR) feature space does not lead to a significant improvement of accuracy. Even application of basic syntactic templates like *subject-verb-object* turns out to be inadequate for typical TREC IR tasks (Strzalkowski et al., 1999). Substantial flexibility in selection and adjustment of such templates for a number of NLP tasks is expected to help. A tool for automated treatment of syntactic templates in the form of constituency parse trees would be desirable.

In this study, we develop a tool for high-level *semantic* classification of natural language sentences based on *full syntactic parse trees*. We introduce the operation of syntactic generalization (SG) which takes a pair of parse trees and finds a set of maximal common sub-trees. We tackle semantic classes which appear in information extraction and knowledge integration problems usually requiring deep natural language understanding (Dzikovska et al., 2005; Galitsky, 2003; Banko et al., 2007). One of such problems is opinion mining, in particular detecting sentences or their parts which express self-contained opinion ready to be grouped and shared. We want to separate *informative/potentially useful* opinion sentences like 'The shutter lag of this digital camera is annoying sometimes, especially when capturing cute baby moments' which can serve as recommendations, from uninformative and/or irrelevant opinion expressions such as 'I received the camera as a Christmas present from relatives and enjoyed it a lot.' The former sentence characterizes a parameter of a camera component, and in the latter, one talks about circumstances a person was given a camera as a gift (Fig. 1a).

What kind of syntactic and/or semantic properties can separate these two sentences into distinct classes? We assume that the classification is done in a domain-independent manner, so no knowledge of 'digital camera' domain is supposed to be applied. Both these sentences have sentiments, the semantic difference between them is that in the former sentiment is attached to a parameter of the camera, and in the letter sentiment is associated with the form in which the camera was received by the author. Can the latter sentence be turned into a meaningful one by referring to its particular feature (e.g., by saying '…and enjoyed its LCD a lot')? No, because then its first part ('received as a present') is not logically connected to its second part ('I enjoyed LCD because the camera was a gift'). Hence we observe that in this example belonging to positive and negative classes constitutes somewhat stable patterns.

Learning based on syntactic parse tree generalization is different from *kernel methods* which are non-parametric density estimation techniques that compute a kernel function between data instances (which can include keywords as well as their syntactic parameters), where a kernel function can be thought of as a similarity measure. Given a set of labeled instances, kernel methods determine the label of a novel

* Tel.: +34 800 322 9266; fax: +34 408 376 7514.
E-mail address: boris.galitsky@ebay.com

0952-1976/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.engappai.2012.09.017

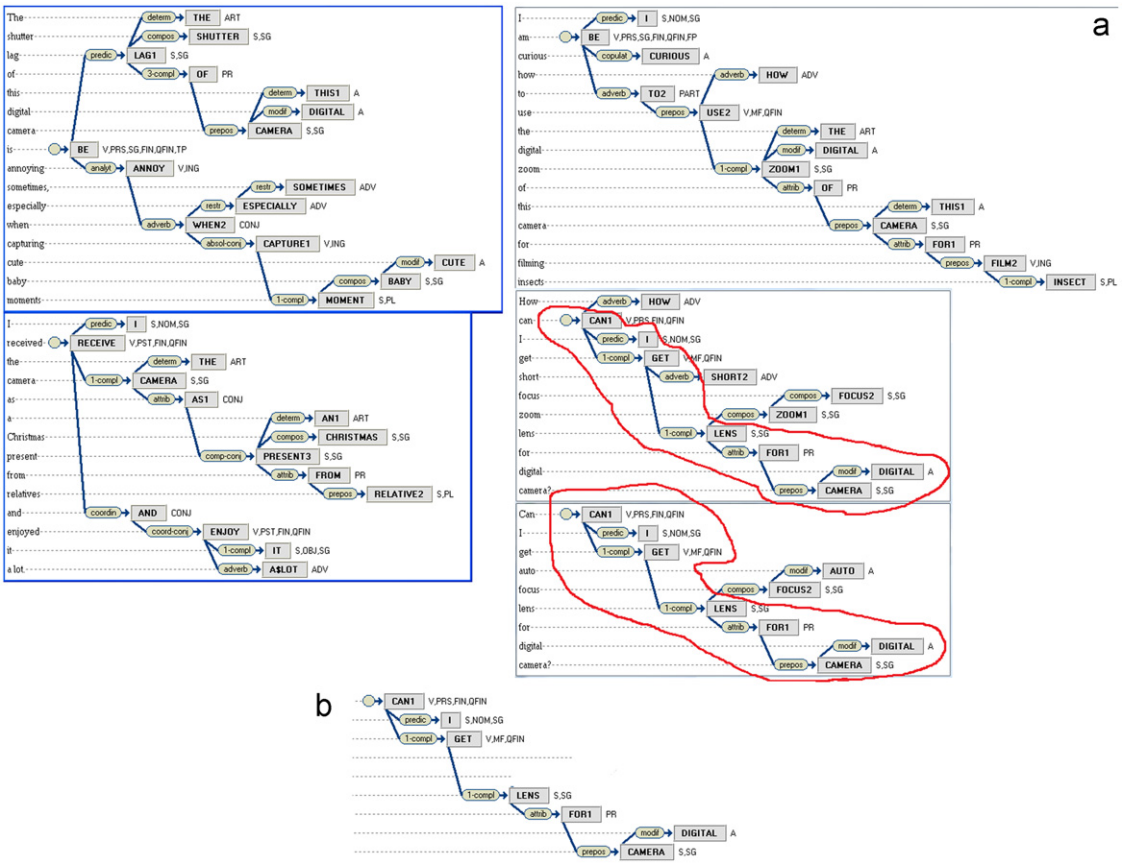


Fig. 1. Syntactic parse tree for informative (on the top, positive class) and uninformative (negative, on the bottom) sentences (on the left). (a) Parse trees for three sentences. The curve shows the common sub-tree (a single one in this case) for the second and third sentence. (b) Generalization results for second and third sentence.

instance by comparing it to the labeled training instances using this kernel function. Nearest neighbor classification and support-vector machines (SVMs) are two popular examples of kernel methods (Fukunaga, 1990; Cortes and Vapnik, 1995). Compared to kernel methods, syntactic generalization (SG) can be considered as structure-based and deterministic; linguistic features retain their structure and are not represented as values.

In this paper we will be finding a set of maximal common sub-tree for a pair of parse tree for two sentences as a measure of similarity between them. It will be done using representation of constituency parse trees via chunking; each type of phrases (NP, VP PRP etc.) will be aligned and subject to generalization.

The main question of this study is whether these semantic patterns can be obtained from complete parse tree structure. Moreover, as we observe the argument structure of how authors communicate their conclusions (as expressed by syntactic structures), they are important for relating a sentence to the above classes. In studies (Galitsky and Kuznetsov, 2008; Galitsky et al., 2009), it was demonstrated that graph-based machine learning can predict plausibility of complaint scenarios based on their argumentation structure. Also, we observed that learning communicative structure of inter-human conflict scenarios can successfully classify the scenarios in a series of domains, from complaint to security-related domains. These findings make us believe that applying similar graph-based machine learning technique to such structure as syntactic trees, which has even weaker links to high-level semantic properties in comparison to these settings, can nevertheless deliver satisfactory classification results.

Most current learning research in NLP employs particular statistical techniques inspired by research in speech recognition, such as hidden Markov models (HMMs) and probabilistic context-free grammars (PCFGs). A variety of learning methods including decision tree and rule induction, neural networks, instance-based methods, Bayesian network learning, inductive logic programming, explanation-based learning, and genetic algorithms can also be applied to natural language problems and can have significant advantages in particular applications (Moreda et al., 2007). In addition to specific learning algorithms, a variety of general ideas from traditional machine learning such as active learning, boosting, reinforcement learning, constructive induction, learning with background knowledge, theory refinement, experimental evaluation methods, PAC learnability, etc., may also be usefully applied to natural language problems (Cardie and Mooney, 1999). In this study we employ nearest neighbor type of learning, which is relatively simple, to focus our investigation on how expressive can similarity between syntactic structures be to detect weak semantic signals. Other more complex learning techniques can be applied, being more sensitive or more cautious, after we confirm that our measure of syntactic similarity between texts is adequate.

The computational linguistics community has assembled large data sets on a range of interesting NLP problems. Some of these problems can be reduced to a standard classification task by appropriately constructing features; however, others require using and/or producing complex data structures such as complete parse trees and operations on them. In this paper we introduce the operation of generalization on the pair of parse tree for two sentences and demonstrate its role in sentence classification. Operation of generalization is defined starting from the level of lemmas to chunks/phrases and all the way to paragraphs/texts.

Please cite this article as: Galitsky, B., Machine learning of syntactic parse trees for search and classification of text. Eng. Appl. Artif. Intel. (2012), http://dx.doi.org/10.1016/j.engappai.2012.09.017

The paper introduces four distinct problems of different complexity where one or another semantic feature has to be inferred from natural language sentences. Then we define syntactic generalization, describe the algorithm and provide a number of examples of SG in various settings including semantic role labeling (SRL). The paper is concluded by the comparative analysis of classification in selected problem domains, search engine description and a brief review of other studies with semantic inference.

Learning syntactic parse trees allows performing semantic inference in a domain-independent manner without using ontologies. At the same time, in contrast to the most semantic inference projects, we will be restricted to a very specific semantic domain (limited set of classes), solving a number of practical problems for the virtual forum platform.

2. SG in search and relevance assessment

In this study we leverage parse tree generalization technique for automation of content management and delivery platform (Galitsky et al., 2011), named integrated opinion delivery environment. This platform combines data mining of web and social networks, content aggregation, reasoning, information extraction, question/answering and advertising to support distributed recommendation forums for a wide variety of products and services. In addition to human users, automated agents answer questions and provide recommendations based on previous postings of human users determined to be relevant. The key technological requirements is based on finding similarity between various kinds of texts, so use of more complex structures representing text meaning is expected to benefit the accuracy of relevance assessment. SG has been deployed at content management and delivery platforms at two portals in Silicon Valley, USA, Datran.com and Zvents.com. We will present evaluation of how the accuracy of relevance assessment has been improved in Evaluation (Section 6).

We focus on four following problems which are essential at various phases of the above application:

1. Differentiating meaningful from meaningless sentences in opinion mining results;
2. Detecting appropriate expressions for automated building of adverts as an advertisement management platform of virtual forums;
3. Classifying user posting in respect to her epistemic state: how well she understands her product needs and how specific is she currently with her product choice;
4. Classifying search results in respect to being relevant and irrelevant to search query.

In all these tasks it is necessary to relate a sentence into two classes, e.g., *informative vs uninformative* opinion, *suitable* vs. *unsuitable, knowledgeable or unknowledgeable user, and relevant/irrelevant answer* to be a basis for advert generation. In both these tasks, decision about belonging to a class cannot be made given occurrence of specific forms; instead, peculiar and implicit linguistic information needs to be taken into account. It is rather hard to formulate and even to imagine classification rules for both of these problems; however finding plentiful examples for respective classes is quite easy. We now outline each of these four problems.

As to the *first* one, traditionally, opinion mining problems is formulated as finding and grouping a set of sentences expressing sentiments about given features of products, extracted from customer reviews of products. A number of comparison shopping sites are now showing such features and the 'strength' of opinions about them as a number of occurrences of such features. However, to increase user confidence and trust in extracted opinion date, it is advisable to link aggregated sentiments for a feature to original quotes from customer reviews; this significantly backs up review-based recommendations by comparative shopping sites.

Among all sentences mentioning the feature of interest, some of them are indeed irrelevant to this feature, does not really express customer opinion about this particular features (and not about something else). For example, 'I don't like touch pads' in reviews on Dell Latitude notebooks does not mean that this touchpad of these notebook series is bad, instead, we have a general customer opinion on a feature which is not expected to be interesting to another user. One can see that this problem for an opinion sentence has to be resolved for building highly trusted opinion mining applications.

We believe this classification problem is rather hard one and require a sensitive treatment of sentence structure, because a difference between meaningful and meaningless sentence with respect to expressed opinion is frequently subtle. A short sentence can be meaningless, its extension become meaningful, but its further extension can become meaningless again. We selected this problem to demonstrate how a very weak semantic signal concealed in a syntactic structure of sentence can be leveraged; obviously, using keyword-based rules for this problem does not seem meaningful.

As to the *second* problem of advert generation, its practical value is to assist business/website manager in writing adverts for search engine marketing. Given the content of a website and its selected landing page, the system needs to select sentences which are most suitable to form an advert.

For example, from the content like

At HSBC **we believe in great loan deals**, **that's why we offer 9.9% APR typical on our loans of $7,500 to $25,000****. It's also why we pledge to pay the difference if you're offered a better deal elsewhere.

What you get with a personal loan from HSBC:

* An instant decision if you're an Online Banking customer and **get your money in hours**, if accepted†
* Our price guarantee: if you're offered a better deal elsewhere we'll pledge to pay you the difference between loan repayments***
* Apply to borrow up to $25,000
* No fees for arrangement or set up
* Fixed monthly payments, so you know where you are
* Optional tailored Payment Protection Insurance.

Please cite this article as: Galitsky, B., Machine learning of syntactic parse trees for search and classification of text. Eng. Appl. Artif. Intel. (2012), http://dx.doi.org/10.1016/j.engappai.2012.09.017

We want to generate the following ads

Great Loan Deals
9.9% APR typical on loans of $7,500 to $25,000. Apply now! Apply for an HSBC loan We offer 9.9% APR typical Get your money in 3 hours!

We show in bold the sentences and their fragments for potential inclusion into an advert line (positive class). This is a semantic IE problem where rules need to be formed automatically (a similar class of problem was formulated in Stevenson and Greenwood, 2005). To form criteria for an expression to be a candidate for an advert line, we will apply SG to the sentences of the collected training sets, and then form templates from the generalization results, which are expected to be much more sensitive than just sets of keywords under traditional keyword-based IE approach.

The *third* problem of classification of epistemic states of a forum user is a more conventional classification problem, where we determine what kind of response a user is expecting:

- general recommendation,
- advice on a series of products, a brand, or a particular product,
- response and feedback on information shared, and others.

For each epistemic state (such as *a new user, a user seeking recommendations, an expert user sharing recommendations, a novice user sharing recommendation*) we have a training set of sentences, each of which is assigned to this state by a human expert. For example (epistemic states are italicized),

"I keep in mind no brand in particular but I have read that Canon makes good cameras"→*user with one brand in mind,* "I have read a lot of reviews but still have some questions on what camera is right for me"→*experienced buyer*. We expect the proper epistemic state to be determined by syntactically closest representative sentence.

Transitioning from keywords match to SG is expected to significantly improve the accuracy of epistemic state classification, since these states can be inferred from the syntactic structure of sentences rather than explicitly mentioned most of times. Hence the results of SGs of the sentences form the training set for each epistemic state will serve as classification templates rather than common keywords among these sentences.

The *fourth* application area of SG is associated with improvement of search relevance by measuring similarity between query and sentences in search results (or snapshots) by computing SG. Such syntactic similarity is important when a search query contains keywords which form a phrase, domain-specific expression, or an idiom, such as "shot to shot time", "high number of shots in a short amount of time". Usually, a search engine is unable to store all of these expressions because they are not necessarily sufficiently frequent, however make sense only if occur within a certain natural language expression.

In terms of search implementation, this can be done in two steps:

1) Keywords are formed from query in a conventional manner, and search hits are obtained by TF*IDF also taking into account popularity of hits, page rank and others.
2) The above hits are filtered with respect to syntactic similarity of the snapshots of search hits with search query. Parse tree generalization comes into play here.

Hence we obtain the results of the conventional search and calculate the score of the generalization results for the query and each sentence and each search hit snapshot. Search results are then re-sorted and only the ones syntactically close to search query are assumes to be relevant and returned to a user.

Let us consider an example of how use of phrase-level match of a query to its candidate answer instead of keywords-based match helps. When a query is relatively complex, it is important to perform match at phrase level instead of keywords level (even taking into account document popularity, TF*IDF, and learning which answers were selected by other users for similar queries previously).

For the following example

http://www.google.com/search?q=how+to+pay+foreign+business+tax+if+I+live+in+the+US

Most of the search results are irrelevant. However, once one starts taking into account the syntactic structure of the query phrases, 'pay-foreign-business-tax', 'I-live-in-US', irrelevant answers where the keywords co-occur in a different way that in the query, are filtered out.

3. Generalizing portions of text

To measure of similarity of abstract entities expressed by logic formulas, a least-general generalization was proposed for a number of machine learning approaches, including explanation based learning and inductive logic programming. Least general generalization was originally introduced by Plotkin (1970). It is the opposite of most general unification (Robinson, 1965) therefore it is also called *anti-unification*. Anti-unification was first studied in Robinson (1965, Plotkin (1970). As the name suggests, given two terms, it produces a more general one that covers both rather than a more specific one as in unification. Let E_1 and E_2 be two terms. Term E is a generalization

Please cite this article as: Galitsky, B., Machine learning of syntactic parse trees for search and classification of text. Eng. Appl. Artif. Intel. (2012), http://dx.doi.org/10.1016/j.engappai.2012.09.017

of E_1 and E_2 if there exist two substitutions σ_1 and σ_2 such that $\sigma_1(E)=E_1$ and $\sigma_2(E)=E_2$. The most specific generalization of $E1$ and $E2$ is called anti-unifier. Here we apply this abstraction to anti-unify such data as text, traditionally referred to as unstructured.

For two words of the same POS, their generalization is the same word with POS. If lemmas are different but POS is the same, POS stays in the result. If lemmas are the same but POS is different, lemma stays in the result.

In this study, to measure similarity between portions of text such as paragraphs, sentences and phrases, we extend the notion of generalization from logic formulas to sets of syntactic parse trees of these portions of text. If it were possible to define similarity between natural language expressions at pure semantic level, least general generalization would be sufficient. However, in horizontal search domains where construction of full ontologies for complete translation from NL to logic language is not plausible, extension of the abstract operation of generalization to syntactic level is required. Rather than extracting common keywords, generalization operation produces a syntactic expression that can be semantically interpreted as a common meaning shared by two sentences.

Let us represent a meaning of two NL expressions by logic formulas and then construct unification and anti-unification of these formulas. Some words (entities) are mapped into predicates, some are mapped into their arguments, and some other words do not explicitly occur in logic form representation but indicate the above instantiation of predicates with arguments. How to express a commonality between the expressions?

- *camera with digital zoom*
- *camera with zoom for beginners*

To express the meanings we use logic predicates *camera*(*name_of_feature, type_of_users*) (in real life, we would have much higher number of arguments), and zoom(*type_of_zoom*). The above NL expressions will be represented as:

camera(*zoom*(*digital*), *AnyUser*)
camera(*zoom*(*AnyZoom*), *beginner*),

where variables (uninstantiated values, not specified in NL expressions) are capitalized. Given the above pair of formulas, unification computes their most general specialization *camera*(*zoom*(*digital*), *beginner*), and anti-unification computes their most specific generalization, *camera*(*zoom*(*AnyZoom*), *AnyUser*).

At syntactic level, we have generalization of two noun phrases as:
{NN-camera, PRP-with, [digital], NN-zoom [for beginners]}.
We eliminate expressions in square brackets since they occur in one expression and do not occur in another. As a result, we obtain
{*NN-camera, PRP-with, NN-zoom*]}, which is a syntactic analog as the semantic generalization above.

Notice that a typical scalar product of feature vectors in a vector space model would deal with frequencies of these words, but cannot easily express such features as co-occurrence of words in phrases, which is frequently important to express a meaning of a sentence and avoid ambiguity.

Since the constituent trees keep the sentence order intact, building structures upward for phrases, we select constituent tree to introduce our phrase-based generalization algorithm. The dependency tree has the word nodes at different levels and each word modifies another word or the root. Because it does not introduce phrase structures, the dependency tree has few nodes than the constituent tree and is less suitable for generalization. Constituent tree explicitly contains word alignment-related information required for generalization at the level of phrases. We use (OpenNLP, 2011) system to derive constituent trees for generalization (chunker and parser). Dependency-tree based, or graph-based similarity measurement algorithms (Bunke, 2003; Galitsky and Kuznetsov, 2008) are expected to perform as well as the one we focus on in this paper.

3.1. Generalizing at various levels: From words to paragraphs

The purpose of an abstract generalization is to find commonality between portions of text at various semantic levels. Generalization operation occurs on the following levels:

- Text
- Paragraph
- Sentence
- Phrases (noun, verb and others)
- Individual word

At each level except the lowest one, individual words, the result of generalization of two expressions is a *set* of expressions. In such set, for each pair of expressions so that one is less general than other, the latter is eliminated. Generalization of two sets of expressions is a set of sets which are the results of pair-wise generalization of these expressions.

We first outline the algorithm for two sentences and then proceed to the specifics for particular levels. The algorithm we present in this paper deals with paths of syntactic trees rather than sub-trees, because it is tightly connected with language phrases. In terms of operations on trees we could follow along the lines of Kapoor and Ramesh (1995).

Being a formal operation on abstract trees, generalization operation nevertheless yields semantic information about commonalities between sentences. Rather then extracting common keywords, generalization operation produces a syntactic expression that can be semantically interpreted as a common meaning shared by two sentences.

1) Obtain parsing tree for each sentence. For each word (tree node) we have lemma, part of speech and form of word information. This information is contained in the node label. We also have an arc to the other node.
2) Split sentences into sub-trees which are phrases for each type: verb, noun, prepositional and others; these sub-trees are overlapping. The sub-trees are coded so that information about occurrence in the full tree is retained.

Please cite this article as: Galitsky, B., Machine learning of syntactic parse trees for search and classification of text. Eng. Appl. Artif. Intel. (2012), http://dx.doi.org/10.1016/j.engappai.2012.09.017

3) All sub-trees are grouped by phrase types.
4) Extending the list of phrases by adding equivalence transformations (Section 3.2).
5) For the set of the pairs of sub-trees for both sentences for each phrase type.
6) For each pair in 5) yield an alignment (Gildea, 2003), and then generalize each node for this alignment. For the obtained set of trees (generalization results), calculate the score.
7) For each pair of sub-trees for phrases, select the set of generalizations with the highest score (least general).
8) Form the sets of generalizations for each phrase types whose elements are sets of generalizations for this type.
9) Filtering the list of generalization results: for the list of generalization for each phrase type, exclude more general elements from lists of generalization for 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 generalization of words occurring in syntactic tree as *word node*. If word forms are different (e.g., one is single and other is plural), then only the lemma of word stays. If the words are different but only parts of speech are the same, the resultant node contains part of speech information only and no lemma. If parts of speech are different, generalization node is empty.

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

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

Generalization is { < *JJ-digital, NN-camera* > , < *NN- today, ADV,Monday* > }, where the generalization for noun phrases is followed by the generalization by adverbial phrase. 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 different sequence with the other generalization nodes.

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

To optimize the calculation of generalization score, we conducted a computational study to determine the POS weights to deliver the most accurate similarity measure between sentences possible (Galitsky et al., 2010). The problem was formulated as finding optimal weights for nouns, adjectives, verbs and their forms (such as gerund and past tense) such that the resultant search relevance is maximum. Search relevance was measured as a deviation in the order of search results from the best one for a given query (delivered by Google); current search order was determined based on the score of generalization for the given set of POS weights (having other generalization parameters fixed). As a result of this optimization performed in (Galitsky et al., 2010), we obtained $W_{NN}=1.0$, $W_{JJ}=0.32$, $W_{RB}=0.71$, $W_{CD}=0.64$, $W_{VB}=0.83$, $W_{PRP}=0.35$ excluding common frequent verbs like *get/take/set/put* for which $W_{\mathrm{VBcommon}}=0.57$. We also set that $W_{<\mathrm{POS},*>}=0.2$ (different words but the same POS), and $W_{<*,\mathrm{word}>}=0.3$ (the same word but occurs as different POSs in two sentences).

Generalization score (or similarity between sentences sent_1, sent_2) then can be expressed as sum through phrases of the weighted sum through words

$\mathrm{word}_{\mathrm{sent1}}$ and $\mathrm{word}_{\mathrm{sent2}}$

$\mathrm{score}(\mathrm{sent}_1, \mathrm{sent}_2)=\sum_{\{\mathrm{NP,\ VP,\ \ldots}\}}\sum \mathrm{W_{POS}}\ \mathrm{word\_generalization}(\mathrm{word}_{\mathrm{sent1}}\ \mathrm{word}_{\mathrm{sent2}})$.

(Maximal) generalization can then be defined as the one with the highest score. This way we define a generalization for phrases, sentences and paragraphs.

Result of generalization can be further generalized with other parse trees or generalization. For a set of sentences, totality of generalizations forms a lattice: order on generalizations is set by the subsumption relation and generalization score. We enforce the associativity of generalization of parse trees by means of computation: it has to be verified and resultant list extended each time new sentence is added. Notice that such associativity is not implied by our definition of generalization.

3.2. Equivalence transformation on phrases

We have manually created and collected from various resources rule base for generic linguistic phenomena. Unlike text entailment system, for our setting we do not need a complete transformation system as long as we have sufficiently rich set of examples. Transformation rules were developed under the assumption that informative sentences should have a relatively simple structure (Romano et al., 2006).

Syntactic-based rules capture entailment inferences associated with common syntactic structures, including simplification of the original parse tree, reducing it into canonical form, extracting embedded propositions, and inferring propositions from non-propositional sub-trees of the source tree (Table 1), see also (Zanzotto and Moschitti, 2006).

Valid matching of sentence parts embedded as verb complements depends on the verb properties, and the polarity of the context in which the verb appears (positive, negative, or unknown). We used a list of verbs for communicative actions from (Galitsky and Kuznetsov, 2008) which indicate positive polarity context; the list was complemented with a few reporting verbs, such as *say* and *announce*, since opinions in the news domain are often given in reported speech, while the author is usually considered reliable. We also used annotation rules to mark negation and modality of predicates (mainly verbs), based on their descendent modifiers.

Please cite this article as: Galitsky, B., Machine learning of syntactic parse trees for search and classification of text. Eng. Appl. Artif. Intel. (2012), http://dx.doi.org/10.1016/j.engappai.2012.09.017

Table 1
Rules of graph reduction for generic linguistic structure. Resultant reductions are italicized.

Category	Original/*Transformed fragment*
Conjunctions	Camera is very stable and *has played an important role in filming their wedding*
Clausal modifiers	Flash was disconnected as *children went out to play in the yard*
Relative clauses	I was forced to close the *LCD*, which *was blinded by the sun*
Appositives	*Digital zoom*, *a feature provided by the new generation of cameras*, dramatically decreases the image sharpness.
Determiners	My customers use their (*an* auto …) auto focus camera for polar expedition (their= > *an*)
Passive	Cell phone can be easily grasped by a hand palm (*Hand palm can easily grasp the cell phone*)
Genitive modifier	Sony's LCD screens work in sunny environment as well as Canon's /(*LCD of Sony… as well as of Canon*)
Polarity	It made me use digital zoom for mountain shots (I *used* digital zoom…)

Important class of transformation rules involves noun phrases. For a single noun group, its adjectives can be re-sorted, as well as nouns except the head one. A noun phrase which is a post-modifier of a head noun of a given phrase can be merged to the latter; sometimes the resultant meaning might be distorted by otherwise we would miss important commonalities between expressions containing noun phrases. An expression 'NP_1 < *of* or *for* > NP_2' we form a single NP with the head noun *head*(NP_2) and *head*(NP_1) playing modifier role, and arbitrary sort for adjectives.

3.3. Simplified example of generalization of sentences

We present an example of generalization operation of two sentences. Intermediate sub-trees are shown as lists for brevity. Generalization of distinct values is denoted by '*'. Let us consider three following 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?

We first draw the parsing trees for these sentences and see how to build their maximal common sub-trees:
Fig. 1(b) Parse trees for three sentences. The curve shows the common sub-tree (a single one in this case) for the second and third sentences.
One can see that the second and third trees are rather similar, so it is straight-forward to build their common sub-tree as an (interrupted) path of the tree (Fig. 1c):
{ MD-can, PRP-I, VB-get, NN-focus, NN-lens, IN-for JJ-digital NN-camera }. At the phrase level, we obtain:

```
Noun pharses: [ [NN-focus NN-* ],  [JJ-digital NN-camera
]]
Verb phrases: [ [VB-get NN-focus NN-* NN-lens IN-for JJ-
digital NN-camera ]]
```

Fig. 1 (c) Generalization results for second and third sentence.
One can see that common words remain in the maximum common sub-tree, except 'can' which is unique for the second sentence, and modifiers for 'lens' which are different in these two sentences (shown as *NN-focus NN-∗ NN-lens*). When sentences are not as similar as sentences 2 and 3, and we proceed to their generalization on phrase-by-phrase basis. Below we express the syntactic parse tree 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),
5(ADJP-curious), 5(JJ-curious),
13(SBAR-how to use the digital zoom of this camera for filming insects), 13(WHADVP-how), 13(WRB-how), 17(S-to use the digital zoom of this camera for filming insects),
17(VP-to use the digital zoom of this camera for filming insects), 17(TO-to),
20(VP-use the digital zoom of this camera for filming insects),
20(VB-use),
24(NP-the digital zoom of this camera), 24(NP-the digital zoom),
24(DT-the),
28(JJ-digital),
36(NN-zoom), 41(PP-of this camera), 41(IN-of), 44(NP-this camera), 44(DT-this),
49(NN-camera), 56(PP-for filming insects), 56(IN-for),
60(NP-filming insects), 60(VBG-filming), 68(NNS-insects)

Please cite this article as: Galitsky, B., Machine learning of syntactic parse trees for search and classification of text. Eng. Appl. Artif. Intel. (2012), http://dx.doi.org/10.1016/j.engappai.2012.09.017

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), 10(VB-get), 14(NP-short focus zoom lens), 14(JJ-
short), 20(NN-focus), 26(NN-zoom), 31(NN-lens),
36(PP-for digital camera), 36(IN-for), 40(NP-digital cam-
era), 40(JJ-digital), 48(NN-camera)]
```

Now we group the above phrases by the phrase type [NP, VP, PP, ADJP, WHADVP. Numbers encode character position at the beginning. Each group contains the phrases of the same type, since the match occurs between the same type.

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]], [], [PP [IN-of DT-this NN-camera], PP [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]], [], [], [SBARQ [WHADVP-How MD-can NP-I VB-get JJ-short NN-focus NN-zoom NN-lens IN-for JJ-digital NN-camera], SQ [MD-can NP-I VB-get JJ-short NN-focus NN-zoom NN-lens IN-for JJ-digital NN-camera]]]`

Sample generalization between phrases:

At the phrase level, generalization starts with finding an alignement between two phrases, where we attempt to set a correspondence between as many words as possible between two phrases. We assure that the alignment operation retains phrase integrity: in particular, two phrases can be aligned only if the correspondence between their head nouns is established. There is a similar integrity constraint for aligning verb, prepositional and other types of phrases (Fig. 2).

Here we show the mapping between either words or respective POS to explain how generalization occurs for each pair of phrases for each phrase type. Six mapping links between phrases correspond to six members of generalization result links. The resultant generalization is shown in bold in the example below for verb phrases VP. We specifically use an example of very different phrases now to demonstrate that although the sentences have the same set of keywords, they are not included in generalization (Fig. 3) because their syntactic occurrence is different.

One can see that that such common concept as '*digital camera*' are automatically generalized from the examples, as well as the verb phrase "*be some-kind-of zoom camera*" which expresses the common meaning for the above sentences. Notice the occurrence of expression [digital-camera] in the first sentence: although *digital* does not refer to *camera* directly, we merge two noun group and *digital*

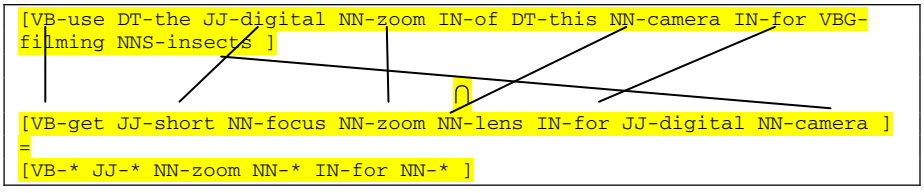


Fig. 2. Alignment between words for two sentences.

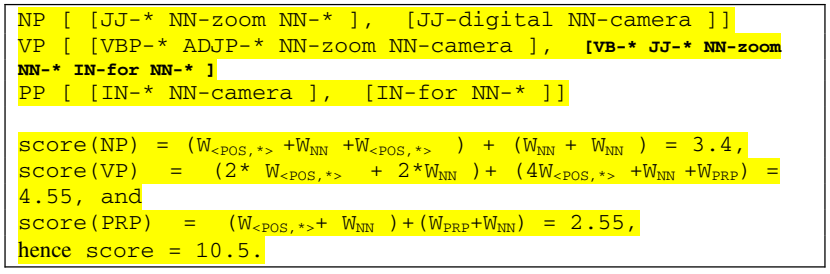


Fig. 3. Generalization results and their score.

Please cite this article as: Galitsky, B., Machine learning of syntactic parse trees for search and classification of text. Eng. Appl. Artif. Intel. (2012), http://dx.doi.org/10.1016/j.engappai.2012.09.017

becomes one of the adjective of this resultant noun group with its head *camera*. It is matched against the noun phrase reformulated in a similar way (but with preposition *for*) from the second sentence with the same head noun *camera*. We present more complex generalization examples in Section 4.

3.4. From syntax to inductive semantics

To demonstrate how the SG allows us to ascend from syntactic to semantic level, we follow Mill's *Direct method of agreement* (*induction*) as applied to linguistic structures. The British philosopher JS Mills, wrote in his 1843 book "A System of Logic": "If *two or more instances* of the phenomenon under investigation have *only one circumstance in common*, the circumstance in which alone all the instances agree, is the *cause* (or *effect*) of the given phenomenon." (Ducheyne, 2008).

Consider a linguistic property A of a phrase *f*. For A to be a necessary condition of some effect E, A must always be present in multiple phrases that deal with E. Therefore, we check whether linguistic properties considered as 'possible necessary conditions' are present or absent in the sentence. Obviously, any linguistic properties which are absent when the meaning is present cannot be necessary conditions for this meaning of a phrase.

For example, the method of agreement can be represented as a phrase f_1 where words {A B C D} occur together with the meaning formally expressed as $< w\ x\ y\ z >$. Consider also another phrase f_2 where words {A E F G} occur together with the same meaning $< w\ t\ u\ v >$ as in phrase f_1. Now by applying generalization to words {A B C D} and {A E F G} we obtain {A} (here, for the sake of example, we ignore the syntactic structure of f_1 and f_2). Therefore, here we can see that word A is the cause of w (has meaning *w*).

Hence we can produce (inductive) semantics applying SG. Semantics cannot be obtained given just syntactic information, however generalizing two or more phrases, we obtain an (inductive) semantic structure. Viewing SG as an inductive cognitive procedure, transition from syntactic to semantic levels can be defined formally. In this work we do not mix syntactic and semantic features to learn a class: instead we derive semantic features from syntactic according to above inductive framework.

3.5. Nearest-neighbor learning of generalizations

To perform classification, we apply a simple learning approach to parse tree generalization results. The simplest decision mechanism can be based on maximizing the score of generalization for an input sentence and a member of the training class. However, to maintain deterministic flavor of our approach we select the nearest neighbor method with limitation for both class to be classified and foreign classes. The following conditions hold when a sentence *U* is assigned to a class R^+ and not to the other class R^-:

1) *U* has a nonempty generalization (having a score above threshold) with a positive example R^+. It is possible that the U has also a nonempty common generalization with a negative example R^-, its score should be below the one for R^+ (This would mean that the graph is similar to both positive and negative examples).
2) For any negative example R^-, if *U* is similar to R^- (i.e., $U * R^- \neq \emptyset$) then *generalization*(U, R^-) should be a sub-tree of *generalization*(U, R^+). This condition introduces the partial order on the measure of similarity. It says that to be assigned to a class, the similarity between the current sentence *U* and the closest (in terms of generalization) sentence from the positive class should be higher than the similarity between *U* and each negative example.

Condition 2 is important to properly handle the nonmonotonic nature of such feature as meaningfulness of an opinion-related sentence. As a sentence gets extended, it can repetitively become meaningless and meaningful over and over, so we need this condition that the parse tree overlap with the foreign class is covered by the parse tree overlap with the true class.

In this project we use a modification of nearest neighbor algorithm to tree learning domain. In our previous studies (Galitsky et al., 2009) we explained why this particular algorithm is better suited to graph data, supporting the learning Explainability feature. We apply a more cautious approach to classification compared to *K*-nearest neighbor, and some examples remain unclassified due to condition 2).

4. Syntactic generalization-based search engine and its evaluation

The search engine based on SG is designed to provide opinions data in an aggregated form obtained from various sources. Conventional search results and Google sponsored link formats are selected as most effective and already accepted by the vast community of users.

4.1. User interface of search engine

The user interface is shown at Fig. 4. To search for an opinion, a user specifies a product class, a name of particular products, a set of its features, specific concerns, needs or interests. A search can be narrowed down to a particular source; otherwise multiple sources of opinions (review portals, vendor-owned reviews, forums and blogs available for indexing) are combined.

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

Automatically generated product advertisement compliant with Google sponsored links format are shown on the right. Phrases in generated advertisements are extracted from original product web pages and possibly modified for compatibility, compactness and appeal to potential users. There is a one-to-one correspondence between products in opinion hits on the left and generated advertisements on the right (unlike in Google, where sponsored links list different websites from those on the left). Both respective business representatives and product users are encouraged to edit and add advertisements, expressing product feature highlights and usability opinions respectively.

Please cite this article as: Galitsky, B., Machine learning of syntactic parse trees for search and classification of text. Eng. Appl. Artif. Intel. (2012), http://dx.doi.org/10.1016/j.engappai.2012.09.017

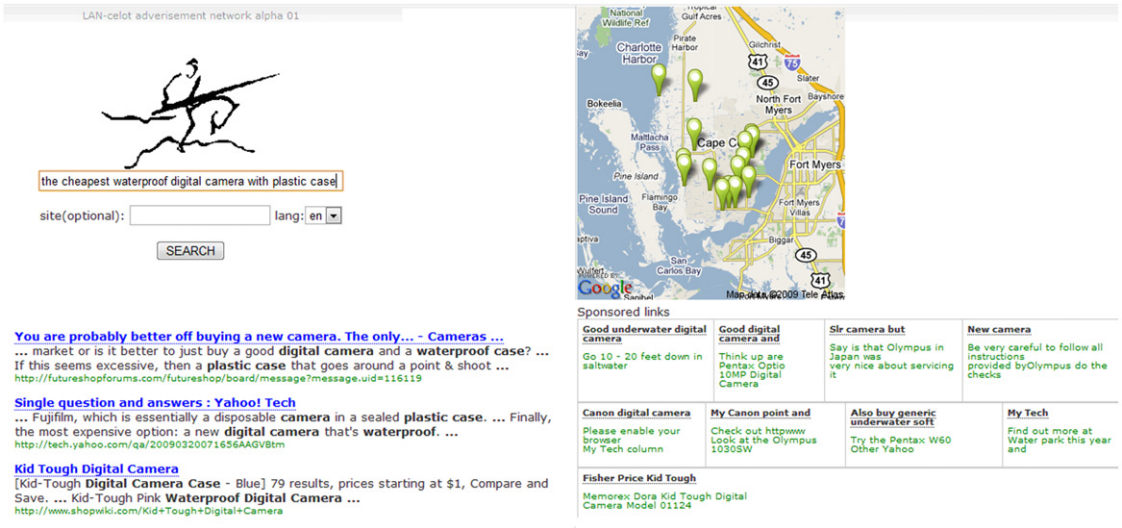


Fig. 4. User interface of generalization-based search engine.

Search phrase may combine multiple sentences, for example: ‘‘I am a beginner user of digital camera. I want to take pictures of my kids and pets. Sometimes I take it outdoors, so it should be waterproof to resist rain’’. Obviously, this kind 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 provides 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. Call anytime.

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

4.2. Qualitative evaluation of search

Obviously, the generalization-based search performance is higher for longer keyword queries and natural language queries, where high sensitivity comparison of query and search results allows finding semantic relevancy between them.

∗We start with the example query ‘‘National Museum of Art in New York’’ (Fig. 5) which illustrates a typical search situation where a user does not know an exact name of an entity. We present the results as ordered by the generalization-based search engine, retaining the information from the original order obtained for this query on Yahoo.com (#x). Notice that the expected name of the museum is either *Metropolitan Museum of Art* or *National Museum of Catholic Art & History*.

The match procedure needs to verify that ‘National’ and ‘Art’ from the query belong to the noun group of the main entity (museum), and this entity is linguistically connected to ‘New York’. If these two conditions are satisfied, we get the first few hits relevant (although mutually inconsistent, it is either museum or academy). As to the Yahoo sort, we can see that first few relevant hits are numbered as #5, #18, #29. Yahoo’s #0 and #1 are on the far bottom of generalization-based search engine, the above condition for ‘National’ and ‘Art’ are not satisfied, so these hits do not seem to be as relevant. Obviously, conventional search engines would have no problems delivering answers when the entity is mentioned exactly (Google does a good job answering the above query; it is perhaps achieved by learning what other people ended up clicking through).

Hence we observe that generalization helps for the queries where important components and linguistic link between them in a query has to be retained in the relevant answer abstracts. Conventional search engines use a high number of relevancy dimensions such as page rank, however for answering more complex questions syntactic similarity expressed via generalization presents substantial benefits.

We perform our quantitative evaluation of search re-ranking performance with two settings:

1) General web search. We do not use machine learning setting here, but instead compute SG score and re-rank online according to this score. We increase the query complexity and observe the contribution of SG
2) Product search in a vertical domain. We analyze various query types and evaluate how automated SG, as well as the one augmented by manually constructed templates, help to improve search relevance.

Please cite this article as: Galitsky, B., Machine learning of syntactic parse trees for search and classification of text. Eng. Appl. Artif. Intel. (2012), http://dx.doi.org/10.1016/j.engappai.2012.09.017

NATIONAL MUSEUM OF CATHOLIC ART & HISTORY - New York, NY **(#5)**
NATIONAL MUSEUM OF CATHOLIC ART & HISTORY - in New York, NY. Get contact info, directions and more at YELLOWPAGES.COM
National Academy Museum & School of Fine Arts**(#18)**
He is currently represented by Ameringer Yohe Fine Art in New York. ... © 2007 National Academy Museum & School of Fine Arts, New York. Join Our Mailing List ...
International Council of Museums: Art Galleries**(#29)**
(In French and English.) **National Museum of** Modern **Art**. Musée du **...** Metropolitan **Museum of Art**, **New York** City. One of the largest **art museums in** the world. **...**
Virtual NYC Tour: New York City Museums**(#23)**
National Museum of the American Indian (**New York** branch) **...** Cloisters is one of the **museums of** the Metropolitan **Museum of Art in New York** City. **...**
Great Museums - SnagFilms**(#9)**
Founded in 1870, the Metropolitan **Museum of Art in New York** City is a three **...** Home Base: The **National** Baseball Hall of Fame and **Museum ...**
National Contemporary Art Museum Gets Seoul Venue**(#2)**
... nearby example is the **National Museum of Art in** Deoksu Palace," said **...** can also refer to the MoMA's (**Museum of** Modern **Art**) annex PS1 **in New York**," he said. **...**
National Lighthouse Museum New York City.com : Arts ...**(#1)**
NYC.com information, maps, directions and reviews on **National** Lighthouse **Museum** and other **Museums in New York** City. NYC.com, the authentic city site, also offer a **...**
National Academy Museum New York City.com : Arts ...**(#0)**
NYC.com information, maps, directions and reviews on **National** Academy **Museum** and other **Museums in New York** City. NYC.com, the authentic city site, also offer a **...**

Fig. 5. Sample search results for generalization-based search engine.

Table 2
evaluation of general web search relevance improvement by SG.

Type of search query	Relevancy of Yahoo search, %, averaging over 10	Relevancy of re-sorting by generalization, %, averaging over 10	Relevancy comp to baseline, %
Three to four word phrases	77	77	100.0
Five to seven word phrases	79	78	98.7
Eight to ten word single sentences	77	80	103.9
Two sentences, >8 words total	77	83	107.8
Three sentences, >12 words total	75	82	109.3

4.3. Evaluation of web search relevance improvement

Evaluation of search included an assessment of classification accuracy for search results as relevant and irrelevant. Since we used the generalization score between the query and each hit snapshot, we drew a threshold of five highest score results as relevant class and the rest of search results as irrelevant. We used the Yahoo search API and also Bing search API and applied the generalization score to find the highest score hits from first fifty Yahoo and Bing search results (Table 2). We selected 100 queries for each set from the log of searches for eBay products and eBay entertainment, which were phrased as web searches. For each query, the relevance was estimated as a percentage of correct hits among the first ten, using the values: {correct, marginally correct, incorrect}. Evaluation was conducted by the authors.

Third and second rows from the bottom contain classification results for the queries of 3–4 keywords which is slightly more complex than an average one (3 keywords); and significantly more complex queries of 5–7 keywords respectively.

For a typical search query containing 3–4 words SG is not in use. One can see that for a 5–7 word phrases SG deteriorates the accuracy and should not be used. However, for longer queries the results are encouraging (almost 4% improvement), showing a visible improvement over current Yahoo & Bing searches once the results are re-ranked based on SG. Substantial improvement can be seen for multi-sentence queries as well.

4.4. Evaluation of product search

We conducted evaluation of relevance of SG—enabled search engine, based on Yahoo and Bing search engine APIs. This evaluation was based on eBay product search domain, with a particular focus on entertainment/things-to-do related queries. Evaluation set included a wide range of queries, 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. In our evaluation we split the totality of queries 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). The evaluation was conducted by the authors, based on proprietary search quality evaluation logs.

For an individual query, the relevance was estimated as a percentage of correct hits among the first ten, using the values: {correct, marginally correct, incorrect} (compare with). 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. This measure is more suitable for product-related searches delivering multiple products, than Mean Reciprocal Rank (MRR), calculated as $1/n\ \Sigma_{i=1\ldots n}\ 1/rk_i$ where n is the number of questions, and rk_i is the rank of the first correct answer to question i. MRR is used for evaluation of a search for information, which can be contained in a single (best) answer, whereas a product search might include multiple valid answers.

For each type of phrase for queries, we formed a positive set of 2000 correct answers and 10,000 incorrect answers (snippets) for training; evaluation is based on 20 searches. These answers were formed from the quality assurance dataset used to improve existing

Please cite this article as: Galitsky, B., Machine learning of syntactic parse trees for search and classification of text. Eng. Appl. Artif. Intel. (2012), http://dx.doi.org/10.1016/j.engappai.2012.09.017

Table 3
Evaluation of product search with manual relevance rules.

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-ranking by generalization, %, averaging over 20 searches	Relevancy of re-ranking by using generalization and manual relevance templates, %, averaging over 20 searches	Relevancy improvement for generalization with manual rules, compared to baseline (averaged for Bing & Yahoo)
Three to four word phrases	Noun phrase	67.4	65.1	75.3	90.6	1.368
	verb phrase	66.4	63.9	74.3	88.5	1.358
	how-to expression	65.3	62.7	73.0	90.3	1.411
	average	66.4	63.9	74.2	89.8	1.379
Five to ten word phrases	Noun phrase	53.2	54.6	76.3	91.7	1.701
	verb phrase	54.7	53.9	75.3	88.2	1.624
	how-to expression	52.6	52.6	73.2	88.9	1.690
	average	53.5	53.7	74.9	89.6	1.672
Two to ten sentences	One verb one noun phrases	52.3	56.1	72.1	88.3	1.629
	both verb phrases	50.9	52.6	71.8	84.6	1.635
	one sent of how-to type	49.6	50.1	74.5	83.9	1.683
	average	50.9	52.9	72.8	85.6	1.648

production search engine before the current project started. 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 (SG score). The results are shown in Table 3.

The answers we select by SG from our evaluation dataset can be: - a false positive like for example "Which US president conducted the war in IRAQ?" answered by "The rabbit is in the bush".

– a false negative in case it is not available or SG operation with the correct answer failed.

To further improve the product search relevance in eBay setting, we added manually formed templates that are formed to enforce proper matching with popular questions which are relatively complex, such as

see-VB *-JJ -*{movie-NN∪picture-NN∪ film-NN } of-PRP best-JJ {director-NN∪producer-NN∪artist-NN∪academy-NN} award-NN [for-PRP], to match questions with phrases

Recommend me a movie which got academy award for best director
Cannes Film Festival Best director award movie
Give me a movie with National Film Award for Best Producer
Academy award for best picture
Movies of greatest film directors of all time

Totally 235 templates were added, 10–20 per each entertainment category or genre. Search relevance results for manual templates are shown in Table 3 column 6.

One can observe that for rather complex queries, we have 64–67% relevance improvement, using manually coded templates, compared to baseline horizontal product search provided by Yahoo and Bing APIs. Automated relevance learning has 30% improvement over baseline for simpler question, 39% for more complex phrases and 36% for multi-sentence queries.

It is worth comparing our search re-ranking accuracy with other studies of learning parse trees, especially statistical approach such as tree kernels. In the TREC dataset of question, Moschitti (2008) used a number of tree kernels to evaluate the accuracy of re-ranking of Google search results. In Moschitti's approach, questions are classified as relevant or irrelevant based on building a tree kernels from all common sub-trees, and using SVM to build a boundary between the classes. The authors achieved 65% over the baseline (Google in 2008) in a specific domain of definitional questions by using word sequences and parsing results-based kernel. In our opinion these results for an educational domain are comparable with our results of real-world product related queries without manual templates. As we demonstrate in this study, using manual templates in product searches further increase search relevance for complex multi-phrased questions.

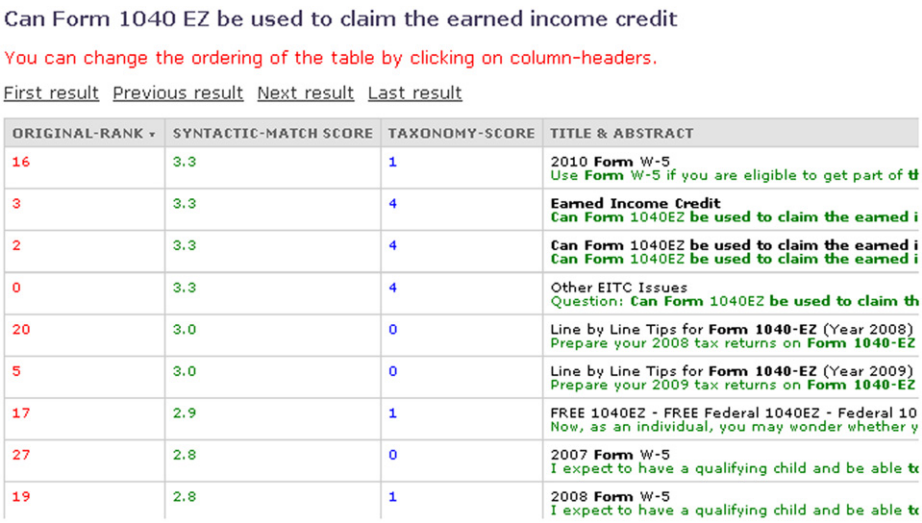
In some learning setting tree kernel approach can provide explicit commonality expressions, similar to the SG approach. Pighin and Moschitti (2009) show the examples of automatically learned commonality expressions for selected classification tasks, which are significantly simpler than commonality structures. Definitional questions from TREC evaluation (Voorhees, 2001) are frequently less ambiguous and better structured than longer queries of real-world users. The maximal common sub-tree are linear structures (and can be reduced to common phrases) such as

president-NN (very specific)
and (VP(VBD)(NP)(PP(IN)(NP)))(very broad).

4.5. Comparison with other means of search relevance improvement

SG was deployed and evaluated in the framework of a Unique European Citizens' attention service (iSAC6+) project, a EU initiative to build a recommendation search engine in a vertical domain. As a part of this initiative, a taxonomy was built to improve search relevance.

Please cite this article as: Galitsky, B., Machine learning of syntactic parse trees for search and classification of text. Eng. Appl. Artif. Intel. (2012), http://dx.doi.org/10.1016/j.engappai.2012.09.017



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 th
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 to
19	2.8	1	2008 **Form** W-5 I expect to have a qualifying child and be able to

Fig. 6. Sorting search results by taxonomy-based and SG scores for a given query "Can Form 1040 EZ be used to claim the earned income credit?".

This taxonomy is used by matching both question and answer to a taxonomy tree and relying on the cardinality of the set of overlapping query terms. The comparison of taxonomy-based score, generalization-based score and the hybrid system score is valuable since the features of various nature are leveraged (pragmatic, syntactic/semantic and hybrid respectively.

We built a tool to perform the comparison of contributions of the above score systems (easy4.udg.edu/isac/eng/index.php). 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 Fig. 6, where 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 and verify the improvement of relevance. An evaluator can sort by original Yahoo score, by SG score, and by taxonomy score, to get a feeling for how each of these scores work and how they correlate with the best order of answers for relevance.

5. Evaluation of text classification problems

5.1. Comparative performance analysis in text classification domains

To evaluate expressiveness and sensitivity of SG operation and associated scoring system, we applied the nearest neighbor algorithm to the series of text classification tasks outlined in Section 2 (Table 4). We form a few datasets for each problem, conduct independent evaluation for this dataset and then average the resultant accuracy (F-measure). Training and evaluation dataset of texts, as well as class assignments, was done by the authors. Half of each set was used for training, and the other half for evaluation; the spilt was random but no cross-validation was conducted. Due to the nature of the problem positive sets are larger than negative sets for sensible/meaningless & ad line problems. For epistemic state classification, negative set includes all other epistemic states or no state at all.

For digital camera reviews, we classify each sentence with respect to sensible/meaningless classes by two approaches:

- A baseline WEKA C4.5, as a popular text classification approach
- SG-based approach.

We demonstrate that a traditional text classification approach poorly handles such a complex classification task, in particular due to slight differences between phrasings for these classes, and the property of non-monotonicity. Using SG instead of WEKA C4.5 brought us 16.1% increase in F-measure for the set of digital camera reviews. In other domains in Table 4, being more traditional for text classification, we do not expect as dramatic improvement (not shown).

Rows 4–7 contain classification data for the reviews on different products, and variability in accuracies can be explained by various levels of diversity in phrasings. For example, the ways people express their feelings about cars is much more diverse than that of about kitchen appliances. Therefore, accuracy of the former task is lower than that of the latter. One can see that it is hard to form verbalized rules for the classes, and hypotheses are mostly domain-dependent; therefore, substantial coverage of varieties of phrasing is required.

To form the training set for ad lines information extraction, we collected positive examples from existing Google ads, scraping more than 2000 ad lines. Precision for extraction of such lines for the same five categories of products is higher than the one for the above tasks of sensible/meaningless classes. A the same time recall of the former is lower than that of the latter, and resultant F-measure is slightly higher for ad lines information extraction, although the complexity of problem is significantly lower. In can be explain of rather high variability of acceptable ad lines ('sales pitches') which have not been captured by the training set.

Overall recognition accuracy of epistemic state classification is higher than for the other two domains because manually built templates for particular states cover a significant portion of cases. At the same time, recognition accuracy for particular epistemic states significantly varies from state to state and was mostly determined by how well various phrasings are covered in the training dataset. We used the same set of reviews as we did for evaluation of meaningless sentences classification and manually selected sentences where the epistemic state of interest was explicitly mentioned or can be unambiguously inferred. For evaluation dataset, we recognized which epistemic state exists in each of 200 sentences. Frequently, there are two or more of such states (without contradictions) per sentence.

Please cite this article as: Galitsky, B., Machine learning of syntactic parse trees for search and classification of text. Eng. Appl. Artif. Intel. (2012), http://dx.doi.org/10.1016/j.engappai.2012.09.017

Table 4
Accuracies of text classification problems.

Problem domain	Dataset	Data set size (# pos/#neg in each of two classes)	Precision relating to a class, (%)	Recall relating to a class, (%)	*F*-measure (%)
Sensible/meaningless	digital camera reviews/processed by WEKA C4.5	120/40	58.8	54.4	56.5
	digital camera reviews	120/40	58.8	74.4	65.6
	cell phone reviews	400/100	62.4	78.4	69.5
	laptop reviews	400/100	74.2	80.4	77.2
	kitchen appliances reviews	400/100	73.2	84.2	78.3
	auto reviews	400/100	65.6	79.8	72.0
Averages for sensible/meaningless performed by SG			65.5	75.3	69.9
Good for ad line/inappropriate for ad line	digital camera webpages	2000/1000	88.4	65.6	75.3
	wireless services webpages	2000/1000	82.6	63.1	71.6
	laptop webpages	2000/1000	69.2	64.7	66.9
	auto sales webpages	2000/1000	78.5	63.3	70.1
	kitchen appliances webpages	2000/1000	78.0	68.7	73.1
Averages for appropriateness for advert line recognition			79.3	65.1	71.4
Epistemic state:	Beginner	30/200	77.8	83.5	80.6
	User with average experience	44/200	76.2	81.1	78.6
	Pro or semi-pro user	25/200	78.7	84.9	81.7
	Potential buyer	60/200	73.8	83.1	78.2
	Open-minded buyer	55/200	71.8	79.6	75.5
	User with one brand in mind	38/200	74.4	81.9	78.0
Averages for epistemic state recognition			75.5	82.4	78.7

Table 5
Improving the precision of text similarity.

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
SG	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 SG and Taxonomy-based	17.2	16.6	17.5	24.1	20.2	18.0

Note also that epistemic states overlap. Low classification accuracy occurs when classes are defined approximately and the boundary between them are fuzzy and beyond expressions in natural language. Therefore we observe that SG gives us some semantic cues which would be hard to obtain at the level of keywords or superficial parsing.

5.2. Example of recognizing meaningless sentences

We use two sorts of training examples to demonstrate typical classes of meaningless sentences which express customer opinions. The first class is specific to the expression of the type $<$entity – sentiment – for – *possible_feature*$>$**.** In most cases, this *possible_feature* is related to entity, characterizes it. However, in this particular case, in the sentence 'For the remainder of the trip the camera was just fine; not even a crack or scratch.'

Here *possible_feature*='remainder of the trip' which is *not* a feature of entity='camera' so we want all sentences similar to this one to be classified as meaningless. To obtain a hypothesis for that, we generalize the above phrase with a sentence like 'For the whole trip we did not have a chance to use this nice camera':

{ [for – DT – trip], [camera]}

The latter sentence can be further generalized with 'I bought Sony in Walwart but did not use this adorable thing'. We obtain {[not – use]} which gives a new meaning of meangless sentences, where an entity *was not used* and therefore sentiment is irrelevant.

What is important for classification is that generalizations obtained from negative examples are not annihilated in positive examples such as ''I could not use the camera', so the expected positive hypothesis will include {[sentiment – NN](NN=entity)} where 'could not use' as a subtree should be substituted as $<$sentiment$>$ placeholder. Hence the generalization of the sentence to be classified 'I didn't have time to use the Canon camera which is my friend's with the above negative hypothesis is not a subsumption of (empty) generalization with the above positive hypothesis.

As one can see, the main barrier to high classification accuracy is the property that the meaningless is not monotonic with respect to growing sentence complexity. A short sentence 'I liked the Panasonic camera' is meaningful, its extension 'I liked the Panasonic camera as a gift of my friend' is not because the sentiment is now associated with *gift.* The further extension of this sentence 'I liked the Panasonic camera as a gift of my friend because of nice zoom' is meaningful again since *nice zoom* is informative.

This case of montonicity can be handled by nearest neighbor learning with moderate success, and it is a very hard case for kernel-based methods because a positive area occurs inside a negative area in turn surrounded by a broader positive area; therefore it can't be separated by hyperplanes, so non-linear SVM kernels would be required (which is not a typical case for text classification types of SVM).

Please cite this article as: Galitsky, B., Machine learning of syntactic parse trees for search and classification of text. Eng. Appl. Artif. Intel. (2012), http://dx.doi.org/10.1016/j.engappai.2012.09.017

5.3. Commercial evaluation of text similarity improvement

We subject the proposed technique of taxonomy-based and SG-based techniques in the commercial main of news analysis at AllVoices.com. The task is to cluster relevant news together, by means of text relevance analysis. By definition, multiple news articles belong to the same cluster, if there is a substantial overlap of involved entities such as geo locations and names of individuals, organizations and other agents, as well as relations between them. Some of these can be extracted by entity taggers, and/or by using taxonomies built offline, and some are handled in real time using SG. The latter is applicable if there is a lack of prior entity information.

In addition to forming a cluster of relevant documents, it is necessary to aggregate relevant images and videos from different sources such as Google image, YouTube and Flickr, and access their relevance given their textual descriptions and tags, where the similar taxonomy and SG-based technique is applied.

Precision of text analysis is achieved by site usability (click rate) by more than nine million unique visitors per month. Recall is accessed manually; however the system needs to find at least a few articles, images and videos for each incoming article. Usually, for web mining and web document analysis recall is not an issue, it is assumed that there is a high number of articles, images and videos on the web for mining.

Relevance is assured by two steps. First, we form a query to image/video/blog search engine API, given event title and first paragraph, extracting noun phrases and filtering them by certain significance criteria. Second, we apply similarity assessment to returned texts for images/videos/blogs and make sure substantial common noun, verb or prepositional sub-phrases can be identified between the seed events and found media.

Precision data for the relevance relation between an articles and other article, blog posting, image and vides is presented in Table 5. Notice that the taxonomy-based method on its own has a very low precision and does not outperform the baseline of the statistical assessment. However, there is a noticeable improvement of precision in hybrid system, where major contribution of SG is improved by a few percents by taxonomy-based method (Galitsky et al., 2011). We can conclude that SG and taxonomy-based methods (which also rely on SG) use different sources of relevance information, so they are indeed complementary to each other.

The objective of SG is to filter out false-positive relevance decision, made by statistical relevance engine, which has been designed following (Liu and Birnbaum, 2007, 2008). The percentage of false-positive news stories was reduced from 29 to 17 (about 30,000 stories/month viewed by 9 million unique users), and the percentage of false positive image attachment was reduced from 24 to 20 (about 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 we need to identify the relevant ones.

The accuracy of our structured learning approach is worth comparing with the other parse tree learning approach based on statistical learning of SVM. Moschitti (2009) compares performances of bag-of-words kernel, syntactic parse trees and predicate argument structures kernel, as well as semantic role kernel and confirms the accuracy improves in this order and reaches F-measure of 68% on TREC dataset. Structured learning methods are better suited for performance-critical production environments serving hundreds millions of users because it better fits modern software quality assurance methodologies. Logs with found commonality expressions are maintained and tracked which assures required performance as system evolves in time and text classification domains change.

6. Related work

Most work in automated semantic inference from syntax deals with much lower semantic level that semantic classes we manage in this study. de Salvo Braz et al. (2005) present a principled, integrated approach to *semantic entailment*. The authors developed an expressive knowledge representation that provides a hierarchical encoding of structural, relational and semantic properties of the text and populated it using a variety of machine learning based tools. An inferential mechanism over a knowledge representation that supports both abstractions and several levels of representations allowed them to begin to address important issues in abstracting over the variability in natural language. Certain reasoning patterns from this work are implicitly implemented by parsing tree matching approach proposed in the current study.

Notice that the set of semantic problems addressed in this paper is of a much higher semantic level compared to SRL, therefore more sensitive tree matching algorithm is required for such semantic level. In terms of this study, semantic level of classification classes is much higher than the level of semantic role labeling or semantic entailment. SLR does not aim to produce complete formal meanings, in contrast to our approach. Our classification classes such as meaningful opinion, proper extraction, and relevant/irrelevant search result are at rather high semantic level, however, cannot be fully formalized; it is hard to verbalize criteria even for human experts.

Usually, classical approaches to semantic inference rely on complex logical representations. However, practical applications usually adopt shallower lexical or lexical-syntactic representations, but lack a principled inference framework. (Bar-Haim et al., 2005) proposed a generic semantic inference framework that operates directly on syntactic trees. New trees are inferred by applying entailment rules, which provide a unified representation for varying types of inferences. Rules are generated by manual and automatic methods, covering generic linguistic structures as well as specific lexical-based inferences. The current work deals with syntactic tree transformation in the graph learning framework (compare with Chakrabarti and Faloutsos, 2006; Kapoor and Ramesh, 1995), treating various phrasings for the same meaning in a more unified and automated manner.

Traditionally, semantic parsers are constructed manually, or are based on manually constructed semantic ontologies, but these are is too delicate and costly. A number of supervised learning approaches to building formal semantic representation have been proposed (Zettlemoyer and Collins, 2005). Unsupervised approaches have been proposed as well, however they applied to shallow semantic tasks (e.g., paraphrasing (Lin and Pantel, 2001), information extraction (Banko et al., 2007), and semantic parsing (Domingos and Poon, 2009)). The problem domain in the current study required much deeper handling syntactic peculiarities to perform classification into semantic classes. In terms of learning, our approach is closer in merits to unsupervised learning of complete formal semantic representation. Compared to semantic role labeling (Carreras and Marquez, 2004) and other forms of shallow semantic processing, our approach maps text to formal meaning representations, obtained via generalization.

Please cite this article as: Galitsky, B., Machine learning of syntactic parse trees for search and classification of text. Eng. Appl. Artif. Intel. (2012), http://dx.doi.org/10.1016/j.engappai.2012.09.017

In the past, unsupervised approaches have been applied to some semantic tasks. For example, DIRT (Lin and Pantel, 2001) learns paraphrases of binary relations based on distributional similarity of their arguments; TextRunner (Banko et al., 2007) automatically extracts relational triples in open domains using a self-trained extractor; SNE applies relational clustering to generate a semantic network from TextRunner triples (Kok and Domingos, 2008). While these systems illustrate the promise of unsupervised methods, the semantic content they extract is nonetheless shallow and we believe it is insufficient for the benchmarking problems presented in this work.

A number of semantic-based approaches has been suggested for problems similar to the four ones used for evaluation in this work. Lamberti et al. (2009) proposed a relation-based page rank algorithm to augment Semantic Web search engines employs data extracted from user query and annotated resource. Relevance is measured as the probability that retrieved resource actually contains those relations whose existence was assumed by the user at the time of query definition. In this study we demonstrated how such problem as search results ranking can be solved based on semantic generalizations based on *local* data—just queries and hit snapshots.

Statistical learning has been applied to syntactic parse trees as well. Statistical approaches are generally based on stochastic models (Zhang et al., 2008). Given a model and an observed word sequence, semantic parsing can be viewed as a pattern recognition problem and statistical decoding can be used to find the most likely semantic representation.

Convolution kernels are an alternative to the explicit feature design which we perform in given paper. They measure similarity between two syntactic trees in terms of their sub-structures (e.g., Collins and Duffy, 2002). These approaches use embedded combinations of trees and vectors (e.g., all vs all summation, each tree and vector of the first object are evaluated against each tree and vector of the second object) and have given optimal results (Moschitti et al., 2006) handling the semantic rolling tasks. For example, given the question "What does S.O.S stand for?", the following representations are used, where the different trees are: the question parse tree, the bag-of-words tree, the bag-of-POS-tags tree and the predicate argument tree

1. (SBARQ (WHNP (WP What))(SQ (AUX does)(NP (NNP S.O.S.))(VP (VB stand)(PP (IN for)));
2. (What *)(does *)(S.O.S. *)(stand *)(for *)(? *);
3. (WP *)(AUX *)(NNP *)(VB *)(IN *)(. *);
4. (ARG0 (R-A1 (What *)))(ARG1 (A1 (S.O.S. NNP)))(ARG2 (rel stand)).

Although statistical approaches will most likely find practical application, we believe that currently structural machine learning approaches will give a more explicit insight on important featured of syntactic parse trees.

Web-based metrics that compute the semantic similarity between words or terms (Iosif and Potamianos, 2009) are complementary to our measure of similarity. The fundamental assumption is used that similarity of context implies similarity of meaning, relevant web documents are downloaded via a web search engine and the contextual information of words of interest is compared (context-based similarity metrics). It is shown that context-based similarity metrics significantly outperform co-occurrence based metrics, in terms of correlation with human judgment.

7. Conclusions

In this study we demonstrated that such high-level sentences semantic features as *being informative* can be learned from the low level linguistic data of complete parse tree. Unlike the traditional approaches to *multilevel* derivation of semantics from syntax, we explored the possibility of linking low level but detailed syntactic level with high-level pragmatic and semantic levels *directly*.

For a few decades, most approaches to NL semantics relied on mapping to First Order Logic representations with a general prover and without using acquired rich knowledge sources. Significant development in NLP, specifically the ability to acquire knowledge and induce some level of abstract representation is expected to support more sophisticated and robust approaches. A number of recent approaches are based on shallow representations of the text that capture lexico-syntactic relations based on dependency structures and are mostly built from grammatical functions extending keyword matching (Durme et al., 2003). Such semantic information as WordNet's lexical chains (Moldovan et al., 2003) can slightly enrich the representation. Learning various logic representations (Thompson et al., 1997) is reported to improve accuracy as well. (de Salvo Braz et al., 2005) makes global use of a large number of resources and attempts to develop a flexible, hierarchical representation and an inference algorithm for it. However, we believe neither of these approaches reaches the high semantic level required for practical application.

Moschitti, 2008 proposed several kernel functions to model parse tree properties in kernel-based machines such as perceptrons or support vector machines. In this study, instead of tackling a high dimensional space of features formed from syntactic parse trees, we apply a more structural machine learning approach to learn syntactic parse trees themselves, measuring similarities via sub-parse trees and not distances in this space. The authors define different kinds of tree kernels as general approaches to feature engineering for semantic role labeling (SLR), and experiments with such kernels to investigate their contribution to individual stages of an SRL architecture both in isolation and in combination with other traditional manually coded features. The results for boundary recognition, classification, and re-ranking stages provide systematic evidence about the significant impact of tree kernels on the overall accuracy, especially when the amount of training data is small. Structure-based methods of this study can leverage limited amount of training cases too.

Tree kernel method assumes we are dealing with arbitrary trees. In this study we are interested in properties of linguistic parse trees, so the method of matching is specific to them. We use the tree rewrite rules specific to parse trees, significantly reducing the dimension of feature space we operate with. In our other studies Galitsky et al., 2011) we used ontologies, further reducing the size of common subtrees. Table 6 performs the further comparative analysis of tree kernel and SG approaches:

Structural method allows combining learning and rule-based approaches to improve the accuracy, visibility and explainability of text classification. Explainability of machine learning results is a key feature in industrial environment. Quality assurance personnel should be able to verify the reason for every decision of automated system. Visibility shows all intermediate generalization results, which allows tracking of how class separation rules are built at each level (pair-wise generalization, generalization ^ sentence, generalization ^ generalization,

Please cite this article as: Galitsky, B., Machine learning of syntactic parse trees for search and classification of text. Eng. Appl. Artif. Intel. (2012), http://dx.doi.org/10.1016/j.engappai.2012.09.017

Table 6
Comparative analysis of two approaches to parse tree learning.

Feature/Approach	Tree Kernels SVM-based	SG based
Phrase rewriting and normalization	Not applied and is expected to be handled by SVM	Rewriting patterns are obtained from literature. Rewriting/normalization significantly reduces the dimension of learning.
Handling semantics	Semantic features are extracted and added to feature space for syntactic features.	Semantics is represented as logic forms. There is a mechanism to build logic forms from generalizations.
Expressing similarity between phrases, sentences, paragraphs	Distance in feature space	Maximal common sub-object, retaining all common features: sub-phrase, sub-sentence, sub-paragraph
Ranking search results	By relevance score, classifying into two classes: *correct and incorrect answers*	By score and by finding entities
Integration with logic form-based reasoning components	N/A	Results of generalization can be fed to a default reasoning system, abduction/inductive reasoning system like JSM (Galitsky et al., 2007), domain-specific reasoning system like reasoning about actions
Combining search with taxonomy	Should be a separate taxonomy-based relevance engine	SG operation is naturally combined with taxonomy tree matching operation (Galitsky et al., 2011)
Using manually formed relevance rules	Should be a separate component, impossible to alter SVM feature space explicitly	Relevance rules in the form of generalizations can be added, significantly reducing dimension of feature space where learning occurs.

(generalization ^ generalization) ^ generalization, etc.) Among the disadvantages of SVM (Suykens et al., 2003) are a lack of transparence of results: it is hard to represent the similarity as a simple parametric function, since the dimension of feature space is rather high. Overall, a tree kernel approach can be thought as statistical AI, and proposed approach follows along the line of logical AI traditionally applied in linguistics two-three decades ago.

Parsing & chunking (conducted by OpenNLP) followed by SG are significantly slower than other operations in a content management system and comparable with operations like duplicate search. Verifying relevance, application of SG should be preceded by statistical keyword-based methods. In real time application components, such as search, we use conventional TF*IDF based approach (such as SOLR/Lucene) to find a set of candidate answers of up to 100 from millions of documents and then apply SG for each candidate. For off-line components, we use parallelized map/reduce jobs (Hadoop) to apply parsing and SG to large volumes of data. This approach allowed a successful combination of efficiency and relevance for serving more than 10 million unique site users monthly at datran.com/allvoices.com, zvents.com and ebay.com.

Proposed approach is tolerant to errors in parsing. For more complex sentences where parsing errors are likely, using OpenNLP, we select multiple versions of parsings and their estimated confidence levels (probabilities). Then we cross-match these versions and if parsings with lower confidence levels provide a higher match score, we select them.

In this study we manually encoded paraphrases for more accurate sentence generalizations. Automated unsupervised acquisition of paraphrase has been an active research field in recent years, but its effective coverage and performance have rarely been evaluated. Romano et al. (2006) proposed a generic paraphrase-based approach for a specific case such as relation extraction to obtain a generic configuration for relations between objects from text. A need for novel robust models for matching paraphrases in texts, which should address syntactic complexity and variability. We believe the current study is a next step in that direction.

Similarly to the above studies, we address the semantic inference in a domain-independent manner. At the same time, in contrast to most semantic inference projects, we narrow ourselves to a very specific semantic domain (limited set of classes), solving a number of practical problems for the virtual forum platform. Learned structures would significantly vary from one semantic domain to another, in contract to general linguistic resources designed for horizontal domains.

Complexity of SG operation is constant. Computing relation $\Gamma_2 \leq \Gamma_1$ for arbitrary graphs Γ_2 and Γ_1 is an NP-complete problem (since it is a generalization of the subgraph isomorphism problem from (Garey and Johnson, 1979)). Finding $X * Y=Z$ for arbitrary X, Y, and Z is generally an NP-hard problem. In (Ganter and Kuznetsov, 2001) a method based on so-called projections was proposed, which allows one to establish a trade-off between accuracy of representation by labeled graphs and complexity of computations with them. Pattern structures consist of objects with descriptions (called patterns) that allow a semilattice operation on them. Pattern structures arise naturally from ordered data, e.g., from labeled graphs ordered by graph morphisms. It is shown that pattern structures can be reduced to formal contexts; in most cases processing the former is more efficient and obvious than processing the latter. Concepts, implications, plausible hypotheses, and classifications are defined for data given by pattern structures. Since computation in pattern structures may be intractable, approximations of patterns by means of projections are introduced. It is shown how concepts, implications, hypotheses, and classifications in projected pattern structures are related to those in original ones.

In particular, for a fixed size of projections, the worst-case time complexity of computing operation * and testing relation $\leq$ becomes constant. Application of projections was tested in various experiments with chemical (molecular) graphs (Kuznetsov and Samokhin, 2005) and conflict graphs (Galitsky et al., 2009). As to the complexity of tree kernel algorithms, they can be run in linear average time (Moschitti, 2008) $O(m+n)$, where m and n are number of nodes in a first and second trees.

Using semantic information for query ranking has been proposed in Aleman-Meza et al. (2003), Ding et al. (2004). However, we believe the current study is a pioneering one in deriving semantic information required for ranking from syntactic parse tree *directly*. In our further studies we plan to proceed from syntactic parse trees to higher semantic level and to explore applications which would benefit from it.

Acknowledgements

We are grateful to our colleagues SO Kuznetsov, B Kovalerchuk and others for valuable discussions, to the anonymous reviewers for their suggestions.

Please cite this article as: Galitsky, B., Machine learning of syntactic parse trees for search and classification of text. Eng. Appl. Artif. Intel. (2012), http://dx.doi.org/10.1016/j.engappai.2012.09.017

Appendix. Implementation of OpenNLP Similarity component

This component does text relevance assessment, accepting two portions of texts (phrases, sentences, paragraphs) and returns a similarity score. https://svn.apache.org/repos/asf/opennlp/sandbox/opennlp-similarity

Similarity component can be used on top of search to improve relevance, computing similarity score between a question and all search results (snippets).

Also, this component is useful for web mining of images, videos, forums, blogs, and other media with textual descriptions. Such applications as content generation and filtering meaningless speech recognition results are included in the sample applications of this component. Relevance assessment is based on machine learning of syntactic parse trees (constituency trees). The similarity score is calculated as the size of all maximal common sub-trees for sentences from a pair of texts.

The objective of Similarity component is to give an application engineer as tool for text relevance which can be used as a black box, no need to understand computational linguistics or machine learning.

First use case of Similarity component: Search

To start with this component, please refer to `SearchResultsProcessorTest.java in package opennlp.tools.similarity.apps`
`public void testSearchOrder()` runs web search using Bing API and improves search relevance.
Look at the code of
`public List < HitBase > runSearch(String query)`
and then at
`private BingResponse calculateMatchScoreResortHits(BingResponse resp, String searchQuery)`
which gets search results from Bing and re-ranks them based on computed similarity score.
The main entry to Similarity component is
`SentencePairMatchResult matchRes=sm.assessRelevance(snapshot, searchQuery);`
where we pass the search query and the snapshot and obtain the similarity assessment structure which includes the similarity score.
To run this test you need to obtain search API key from Bing at www.bing.com/developers/s/APIBasics.html and specify it in
`public class BingQueryRunner in`
`protected static final String APP_ID.`

Solving a unique problem: Content generation

To demonstrate the usability of Similarity component to tackle a problem which is hard to solve without a linguistic-based technology, we introduce a content generation component:
`RelatedSentenceFinder.java`
The entry point here is the function call
`hits=f.generateContentAbout(Albert Einstein);`
which writes a biography of Albert Einstein by finding sentences on the web about various kinds of his activities (such as 'born', 'graduate', 'invented' etc.).

The key here is to compute similarity between the seed expression like "Albert Einstein invented relativity theory" and search result like

"Albert Einstein College of Medicine | Medical Education | Biomedical…
www.einstein.yu.edu/Albert Einstein College of Medicine is one of the nation's premier institutions for medical education,…"
and filter out irrelevant search results.
This is done in function

```
public HitBase augmentWithMinedSentencesAndVerifyRelevance(HitBase item, String originalSentence,
List < String > sentsAll)
SentencePairMatchResult matchRes=sm.assessRelevance(pageSentence+title, originalSentence);
```

You can consult the results in gen.txt, where an essay on Einstein bio is written.
These are examples of generated articles, given the article title
www.allvoices.com/contributed-news/9423860/content/81937916 and
www.allvoices.com/contributed-news/9415063

Solving a high-importance problem: Filtering out meaningless speech recognition results.

Speech recognitions SDKs usually produce a number of phrases as results, such as
"remember to buy milk tomorrow from trader joes",
"remember to buy milk tomorrow from 3 to jones"
One can see that the former is meaningful, and the latter is meaningless (although similar in terms of how it is pronounced). We use web mining and Similarity component to detect a meaningful option (a mistake caused by trying to interpret meaningless request by a query understanding system such as Siri for iPhone can be costly).
`SpeechRecognitionResultsProcessor.java does the job:`
`public List < SentenceMeaningfullnessScore > runSearchAndScoreMeaningfulness(List < String > sents)`
re-ranks the phrases in the order of decrease of meaningfulness.
Similarity component internals are in the package `opennlp.tools.textsimilarity.chunker2matcher`

Please cite this article as: Galitsky, B., Machine learning of syntactic parse trees for search and classification of text. Eng. Appl. Artif. Intel. (2012), http://dx.doi.org/10.1016/j.engappai.2012.09.017

ParserChunker2MatcherProcessor.java does parsing of two portions of text and matching the resultant parse trees to assess similarity between
these portions of text.
To run ParserChunker2MatcherProcessor
private static String MODEL_DIR=resources/models; needs to be specified
The key function
public SentencePairMatchResult assessRelevance(String para1, String para2)
takes two portions of text and does similarity assessment by finding the set of all maximum common subtrees
of the set of parse trees for each portion of text
It splits paragraphs into sentences, parses them, obtained chunking information and produces grouped phrases (noun, evrn, prepositional etc.):
public synchronized List<List<ParseTreeChunk>> formGroupedPhrasesFromChunksForPara(String >para)
and then attempts to find common subtrees:
in ParseTreeMatcherDeterministic.java
List<List<ParseTreeChunk>> res=md.matchTwoSentencesGroupedChunksDeterministic(sent1GrpLst, sent2GrpLst)
Phrase matching functionality is in package opennlp.tools.textsimilarity;
ParseTreeMatcherDeterministic.java:
Here's the key matching function which takes two phrases, aligns them and finds a set of maximum common sub-phrase
public List<ParseTreeChunk> generalizeTwoGroupedPhrasesDeterministic
Package structure is as follows:
opennlp.tools.similarity.apps: 3 main applications
opennlp.tools.similarity.apps.utils: utilities for above applications
opennlp.tools.textsimilarity.chunker2matcher: parser which converts text into a form for matching parse trees
opennlp.tools.textsimilarity: parse tree matching functionality.

Comparison with bag-of-words approach

```
// we first demonstrate how similarity expression for DIFFERENT cases have
// too high score for bagOfWords
String phrase1=How to deduct rental expense from income ;
String phrase2=How to deduct repair expense from rental income.;
List<List<ParseTreeChunk>> matchResult=parser.assessRelevance(phrase1,
phrase2).getMatchResult();
assertEquals(
matchResult.toString(),
"[[ [NN-expense IN-from NN-income ],[JJ-rental NN-* ],[NN-income ]], [ [TO-to VB-deduct JJ-rental NN-* ],[VB-deduct NN-expense IN-from NN-income ]]]);
System.out.println(matchResult);
double matchScore=parseTreeChunkListScorer
.getParseTreeChunkListScore(matchResult);
double bagOfWordsScore=parserBOW.assessRelevanceAnd
GetScore(phrase1,
phrase2);
assertTrue(matchScore+2<bagOfWordsScore);
System.out.println(MatchScore is adequate (=+matchScore
+) and bagOfWordsScore=+bagOfWordsScore+ is too high);
// we now demonstrate how similarity can be captured by POS and cannot be
// captured by bagOfWords
phrase1=Way to minimize medical expense for my daughter;
phrase2=Means to deduct educational expense for my son;
matchResult=parser.assessRelevance(phrase1, phrase2).getMatchResult();
assertEquals(
matchResult.toString(),
"[[ [JJ-*NN-expense IN-for PRP$-my NN-*],[PRP$-my NN-*]], [ [TO-to VB-*JJ-*NN-expense IN-for PRP$-my NN-*]]]);
System.out.println(matchResult);
matchScore=parseTreeChunkListScorer
.getParseTreeChunkListScore(matchResult);
bagOfWordsScore=parserBOW.assessRelevanceAndGetScore(phrase1, phrase2);
assertTrue(matchScore>2 * bagOfWordsScore);
System.out.println(MatchScore is adequate (=+matchScore
+ ) and bagOfWordsScore=+bagOfWordsScore+ is too low);
```

Please cite this article as: Galitsky, B., Machine learning of syntactic parse trees for search and classification of text. Eng. Appl. Artif. Intel. (2012), http://dx.doi.org/10.1016/j.engappai.2012.09.017

References

Allen, J.F., 1987. Natural Language Understanding. Benjamin Cummings.

Alessandro Moschitti, 2008. Kernel methods, syntax and semantics for relational text categorization. In: Proceedings of ACM 17th Conference on Information and Knowledge Management (CIKM). Napa Valley, CA.

Banko, Michael J. Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. 2007 Open information extraction from the web. In Proceedings of the 20th International Joint Conference on Artificial Intelligence, pp. 2670–2676, Hyderabad, India: AAAI Press.

Bar-Haim, R., Dagan, I., Greental, I. Shnarch, E., 2005. Semantic Inference at the Lexical-Syntactic Level AAAI-05.

Bunke, H., 2003. Graph-based tools for data mining and machine learning. Lecture Notes Comput. Sci. 2734/2003, 7–19.

Cardie, C., Mooney, R.J., 1999. Machine learning and natural language. Mach. Learn. 1 (5).

Carreras, X., 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.

Collins, M., Nigel, Duffy, 2002. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In: Proceedings of ACL02.

Cortes, C., Vapnik, V., 1995. Support-vector networks. Mach. Learn. 20, 273–297.

de Salvo Braz, R., Girju, R., Punyakanok, V., Roth, D., Sammons, M., 2005. An inference model for semantic entailment in natural language. Proc. AAAI-05.

Domingos, P., Poon, H. 2009.Unsupervised Semantic Parsing, with. In: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, Singapore: ACL.

Ducheyne, Steffen, 2008. J.S. Mill's Canons of induction: from true causes to provisional ones. Hist. Philos. Logic 29 (4), 361–376.

Durme, B.V., Huang, Y., Kupsc, A., Nyberg, E., 2003. Towards light semantic processing for question answering. HLT Workshop Text Meaning.

Dzikovska, M., Swift, M., Allen, J., William de Beaumont, W., 2005. Generic Parsing for Multi-domain Semantic Interpretation. International Workshop on Parsing Technologies (Iwpt05), Vancouver, BC.

Fabio Massimo Zanzotto Alessandro Moschitti, 2006.Automatic learning of textual entailments with cross-pair similarities. In: Proceedings of the Joint 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING-ACL), Sydney, Australia.

Fukunaga, K., 1990. Introduction to Statistical Pattern Recognition Academic Press, San Diego.

Galitsky, B., 2003. Natural Language Question Answering System: Technique of Semantic Headers. Advanced Knowledge International, Australia.

Galitsky, B., Kuznetsov, S.O., 2008. Learning communicative actions of conflicting human agents. J. Exp. Theor. Artif. Intell. 20 (4), 277–317.

Galitsky, B., González, M.P., Chesñevar, C.I., 2009. A novel approach for classifying customer complaints through graphs similarities in argumentative dialogue. Decis. Support Syst. 46-3, 717–729.

Galitsky, B., Dremov, D.A, Kuznetsov, S.O. 2010.Increasing the relevance of meta-search using parse trees. 12th Russian National AI Conference, M., PhysMatLit, 1, 261–266 (in Russian).

Galitsky, B., Josep Lluis de la Rosa, Gabor Dobrocsi (2011) Building Integrated Opinion Delivery Environment. FLAIRS-24, West Palm Beach, FL May 2011.

Ganter, B., Kuznetsov, SO., 2001. Pattern Structures and Their Projections. Conceptual Structures: Broadening the Base, pp. 129–142.

Garey, M.R., Johnson, D.S., 1979. Computers and Intractability. A Guide to the Theory of NP-Completeness, San Francisco, CA: Freeman.

Gildea, D. 2003. Loosely tree-based alignment for machine translation. In: Proceedings of the 41th Annual Conference of the Association for Computational Linguistics (ACL-03), pp. 80–87, Sapporo, Japan.

Iosif, E., Potamianos,A. Unsupervised semantic similarity computation between terms using web documents IEEE Trans. Knowledge Data Eng., 13, 2009.

Kuznetsov, S.O., Samokhin, M.V., 2005. Learning closed sets of labeled graphs for chemical applications. Induc. Logic Program., 190–208.

Lamberti, F., Sanna, Andrea, Demartini, Claudio, 2009. A relation-based page rank algorithm for semantic web search engines. IEEE Trans. Knowledge Data Eng. 21 (1), 123–136.

Lin, D., Pantel, P. 2001. DIRT: discovery of inference rules from text. In: Proceedings of ACM SIGKDD Conference on Knowledge Discovery and Data Mining 2001, pp. 323–328.

Liu, J., Birnbaum, L., 2008. What do they think? Aggregating local views about news events and topics. In: Proceedings of the 17th International Conference on World Wide Web (WWW'08).

Liu, J., Birnbaum, L., 2007. Measuring Semantic Similarity between Named Entities by Searching the Web Directory", In Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence (WI"07).

Moldovan, D., Clark, C., Harabagiu, S., Maiorano, S. 2003. Cogex: A logic prover for question answering. In: Proceedings of HLTNAACL 2003.

Moreda, P., Navarro, B., Palomar, M., 2007. Corpus-based semantic role approach in information retrieval. Data Knowledge Engineer. 61, 467–483.

Moschitti,A. 2006 Efficient convolution kernels for dependency and constituent syntactic trees. In" Proceedings of the 17th European Conference on Machine Learning, Berlin, Germany.

Moschitti, A.2009 Syntactic and semantic kernels for short text pair categorization. In: Proceedings of the 12th Conference of the European Chapter of the ACL.

OpenNLP, 2011. opennlp.apache.org.

Plotkin, G.D., 1970. A note on inductive generalization. in: Meltzer, B., Michie, D. (Eds.), Machine Intelligence, vol. 5. Elsevier, North-Holland, New York, pp. 153–163.

Robinson, J.A., 1965. A machine-oriented logic based on the resolution principle. J. Assoc. Comput. Mach. 12, 23–41.

Romano, L., Kouylekov, M., Szpektor, I., Dagan, I., Lavelli, A. 2006, Investigating a generic paraphrase-based approach for relation extraction. In: Proceedings of EACL, pp. 409–416.

Ravichandran, D, 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.

Stanley, Kok, Pedro, Domingos, 2008. Extracting semantic networks from text via relational clustering. In: Proceedings of the Nineteenth European Conference on Machine Learning (ECML 2008).

Stevenson, M., Greenwood, M.A. 2005.A semantic approach to IE pattern induction. In: Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005), Ann Arbor, MI.

Suykens, J.A.K., Horvath, G., Basu, S., Micchelli, C., Vandewalle, J. (Eds.), 2003. Computer and Systems Sciences, vol. 190. IOS Press NATO-ASI Series III.

T., Strzalkowski, J.P., Carballo, J., Karlgren, A.H.P.,Tapanainen, T., Jarvinen. 1999.Natural language information retrieval: TREC-8 report. In: Text REtrieval Conference.

Thompson, C., Mooney, R., Tang, L. 1997. Learning to parse NL database queries into logical form. In: Workshop on Automata Induction, Grammatical Inference and Language Acquisition.

Voorhees E.M.2004. Overview of the TREC 2001 Question Answering track. In TREC.

Zettlemoyer, Luke S., Michael, Collins, 2005. Learning to map sentences to logical form: structured classification with probabilistic categorial grammars. In: Proceedings of the Twenty First Conference on Uncertainty in Artificial Intelligence (UAI).

Zhang, M., Zhou, G.D., Aw, A., March 2008. Exploring syntactic structured features over parse trees for relation extraction using kernel methods. Inf. Process. Manage.:Int. Journal 44 (2), 687–701.

Please cite this article as: Galitsky, B., Machine learning of syntactic parse trees for search and classification of text. Eng. Appl. Artif. Intel. (2012), http://dx.doi.org/10.1016/j.engappai.2012.09.017