Mapping Syntactic to Semantic Generalizations of Linguistic Parse Trees

Boris Galitsky[1], Josep Lluis de la Rosa[1] & Gábor Dobrocsi[2]

[1] Univ. Girona Spain
bgalitsky@hotmail.com

[2] Univ Miskolc Miskolc Hungary
gadomail@gmail.com

Abstract

We define sentence generalization and generalization diagrams via a special case of least general generalization (LGG) as applied to linguistic parse trees. Similarity measure between linguistic parse trees is developed as LGG operation on the lists of sub-trees of these trees. The diagrams introduced are representation of mapping between the syntactic generalization level and semantic generalization level. Generalization diagrams are intended as a framework to compute semantic similarity between texts relying on linguistic parse tree data. Such structured approach significantly improves text relevance assessment in a horizontal domain, where ontologies are not available.

Introduction

It is hard to overestimate an importance of building semantic representation from syntactic level for natural language understanding. This task has immediate applications in tasks such as information extraction and question answering (Allen 1987, Cardie and Mooney 1999, Ravichandran and Hovy 2002). In the last ten years there has been a dramatic shift in computational linguistics from manually constructing grammars and knowledge bases to partially or totally automating this process by using statistical learning methods trained on large annotated or unannotated natural language corpora. In this study we consider a possibility of proceeding from syntactic parse tree to such level of semantic representation as a specific kind of conceptual graph.

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 & Mooney 1999). In this study we focus our investigation on how expressive can similarity between syntactic structures be to detect weak semantic signals in a domain-independent manner (Galitsky 2003).

We also attempt to approach conceptual graph level (Sowa 1984, Polovina & Heaton 1992) using pure syntactic information such as syntactic parse trees and applying learning to it to increase reliability and consistency of resultant semantic representation. The purpose of such automated procedure is to tackle information extraction and knowledge integration problems usually requiring deep natural language understanding (Dzikovska et al. 2005, Galitsky et al 2010, Banko et al 2007) and cannot be solved at syntactic level. Among such problems are text relevance, and semantic similarity between queries and answers under question answering. Also, having defined semantic similarity, one can perform classification into semantic classes. The purpose of generalization diagrams is to support such classification tasks. We demonstrate how generalization diagrams are constructed in Section 4.

Search and similarity of syntactic parse trees

We apply parse tree generalization technique to solving the problem of classifying search results in respect to being relevant and irrelevant to search query. Our evaluation of matching mechanism is associated with improvement of search relevance by checking syntactic similarity between query and sentences in search hits. 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 taking into account statistical parameters of occurrences these words in documents, popularity of hits, page rank and others.
2) Above hits are filtered with respect to syntactic and semantic similarity of the snapshots of search hits with search query. Generalization diagram is used to compute such similarity at both levels: scoring is based on the size of common maximal sub-tree et respective levels. 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.

Generalizing natural language sentences

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*.

In this study, to measure similarity between natural language (NL) expressions, we extend the notion of generalization from logic formulas to syntactic parse trees of these expressions. 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, therefore 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. How to express a commonality between the expressions?

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

To express the meanings we use predicates *camera(name_of_feature, type_of_users)* (in real life we would hive 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 special generalization, *camera(zoom(AnyZoom), AnyUser).*

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 word one, result of generalization of two expressions is a *set* of expressions. In such set, expressions for which there exist less general expressions are eliminated. Generalization of two sets of expressions is a set of sets which are the results of pair-wise generalization. We first outline the algorithm for two sentences and then proceed to the specifics for particular levels.

Being a formal operation on abstract trees, generalization operation nevertheless yields semantic information about commonalities between sentences. The algorithm is as follows:

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.
3) All sub-trees are grouped by phrase types.
4) Extending the list of phrases by adding equivalence transformations.
5) Generalize each pair of sub-trees for both sentences for each phrase type.
6) For each pair of sub-trees yield an alignment, 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 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 contains {*digital - camera , today – Monday*} , where part of speech information is not shown. *buy* is excluded from both generalizations because it occurs in a different order in the above phrases. *Buy - digital - camera* is not a generalization 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*.

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. Generalization of parse trees obeys the associativity 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.

Constructing generalization diagrams

We now demonstrate how the generalization framework yields generalization diagrams for semantic classification. These diagrams are intended as representation of correspondence between generalizations on syntactic and semantic levels. We use notes from a number of customers of a bank. The dataset of five paragraphs is introduced and then illustrated a step-by-step learning procedure.

1p. A friend transferred funds from a checking to a savings account. He then used the saving funds to pay for his mortgage.

2p. Premier account customers decided to transfer their funds from premier to regular savings account. The couple then used their premier account for automated mortgage payment.

3p. A mortgagee customer transferred the mortgage account from fixed to adjustable. She then decided to use the remaining funds as a last payment of mortgage for her second home.

1n. A broker transferred his title from corporate brokerage to individual accounts. He used to deposit significant personal funds to the brokerage account.

2n. A manager transferred rent from rent collection corporate to investment accounts. In the past he used to deposit rent in his investment brokerage account directly.

To demonstrate a deep level understanding of meanings of these paragraphs, let us introduce two classes of "individual bank users" and "corporate bank users" and demonstrate how these classes can be formed from our data and classification performed. Notice that there is no explicit indication of belonging to one of this classes, it has to be inferred from text. There could be other classes where semantic information has to be inferred such as 'obtained funds are used for something' and 'no such statement is made', 'account type transfer' and

'refinancing', and many more. We use the denotation {1p, 2p, 3p} for the set of positive examples and {1n, 2n} for the set of negative examples. For each example, we enumerate sentences in paragraphs as {a, b, …}.

We intend to express commonalities between the elements of training set to 'explain' belonging to a class, following the classical methodology of induction (Mill 1843). We hypothesize that common linguistic features of a training set *cause* the target feature (the class). In this section we form these features on both syntactic level by means of generalization and on semantic level by means of logical anti-unification. To do that, we will first proceed on syntactic level, and then show how it can be done on semantic level of logic forms. Then we finally show how the syntactic level can be mapped into semantic one.

We build a lattice of generalizations separately for positive and negative sets of paragraphs. The order of generalization is selected such that it results in the maximal lattice in the sense that the total score of all expressions for nodes is the highest. For example, for three paragraphs, it is always higher score to generalize two paragraphs with higher similarity with each other first. Machine learning procedure is then relating a new paragraph to a lattice of training set examples of either class. The learning procedure occurs on both syntactic and semantic levels as well, since either properties may lead to belonging to a class.

The lattice for three paragraphs of the positive set is shown at Fig. 1a. For each lemma in generalization result we use a simplified denotation: either lemma itself (if available) or POS. Single lines depict generalizations for the first sentence of each paragraph (a), double line – for the second sentence (b). There are multiple sentences appearing in different order in a general case. The lattice depicts the relation of "being more general' between generalization results.

Mapping into logic forms

To define mapping into logic forms, we need to form logical predicates and specify semantic types of their arguments. We don't believe that semantic types can be adequately define at text processing time: we would have to keep adding new semantic types as we encounter new co-occurrences of words for predicates and instances for their arguments which has been selected. For the selected domain of financial services, we form the logical predicates enumerated below to represent meanings of entities in our sample paragraphs. These definitions would follow semantic role labeling style (like for 'transfer'), with additional domain-specific constraints on arguments. Additionally, nouns can form logical predicates if they express entities important for the current domain, like *account*. Here we use square brackets for comments:

transfer(who [agent], what [from-what], to-what [result]).

use(who, what [e.g. funds], for-what [for certain purpose]).

account(type [standard account type like checking/saving], attribute [all other account parameters together]).

deposit(who, what [which funds], to-what[account]).

All other logical predicates in this domain just have a single argument for an attribute *mortgage(attribute), customer(attribute), rent(attribute[action with rent), title(attribute[what kind of title]).*

Unlike generalization operation, synonyms are used for building logic forms. Synonyms are very domain specific, for example *account = fund* in expression [VBX-deposit PRP-to]. In other cases (domains), like 'software user accounts', *account* and *fund* cannot be synonyms.

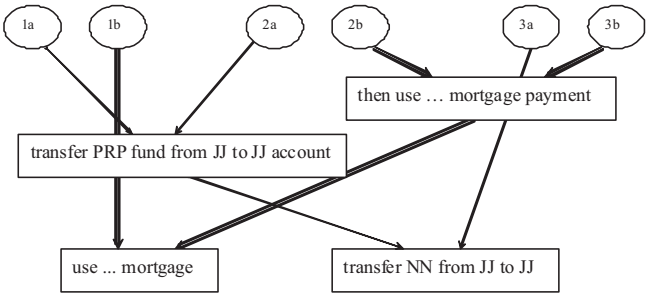

Fig 1a: Generalization diagram for three paragraphs from positive set.

Commonalities between paragraphs of positive class at the semantic level are depicted in Fig. 1b. A lattice shows the order (in terms of generality) between logic forms for original sentences and paragraphs, and also between anti-unifications results for logic forms.

Analogously to the syntactic level, single lines correspond to the first sentence and double line – to the second sentence.

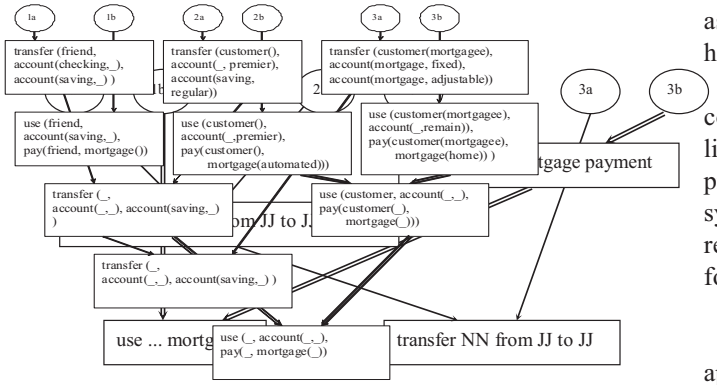

Fig. 1b: Semantic generalization diagram for three paragraphs.

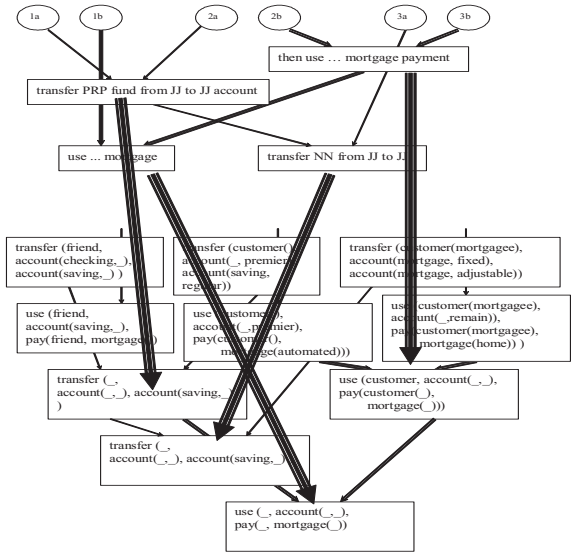

Fig.1c: Generalization diagram showing the mapping between syntactic and semantic

We show the mapping between the syntactic level of Fig. 1a and semantic level of Fig. 1b at Fig. 1c. For a pair sentences, we can first generalize them and then translate result into a logic form. Alternatively, we can translate each sentence into logic form first and then anti-unify these logic forms. Obviously, for simple cases the results are identical, however in general case it is not true. Both operations lead to loss of information of various sorts.

Fig. 1c shows multiple paths to the results of operations of generalization and anti-unification. There is a criterion for optimal path: the resultant score of expression. For a logic form, the score is a number of terms in the expression; this fits well the score of generalization. We define an *optimal path* to the logic form of a set of samples as the one leading to the resultant logic form with the highest score.

Optimality of paths for finding syntactic and semantic commonalities between text paragraphs is grounded in linguistic features. For example, if anti-unification precedes generalization, using such semantic operation as synonym substitution and anaphora resolution makes resultant expressions more complete. If we build logic form from two sentences,

predicate1(customer, ...) [from the first sentence] and

predicate2(he, ...) [from the second sentence], we can apply the obtained fact that 'he' = 'customer' into resultant form

predicate1(customer,...) & predicate2(customer,...).

Otherwise, if we apply generalization first, we would not be able to apply that fact and the resultant logic form will miss the value 'customer'. The reader can see that when generalization results are mapped into 'richer' logic form representation, one of the above semantic operations occurs.

Evaluation

Using generalization, we attempted to improve relevancy of Yahoo! Search, using Boss search API. We used the score of generalization between a query and each hit snapshot, and sorted the hits by this score only. We than evaluated whether search results were improved for complex queries (more than 5 words) with indication of relationships between entities, such as "How to get visa to China in San Francisco".

We use Yahoo! search results as a baseline and evaluated how relevant are first 10 results in Yahoo! search and the results of re-sorting based on generalization presented in this paper. For a set of search results, we measure a portion of those confirmed to be relevant by a selected expert. For initial evaluation, presented in this paper, we used 10 searches for each category. Evaluation environment is publicly available at
http://box.cs.rpi.edu:8080/wise/compare.jsp

Where generalization-based re-sorting of search results is implemented and Yahoo! search result order is shown at the end of snapshot with a number **(#).** Generalization is also designed to handle multi-sentence query (Table 1, last two rows).

We observe that the more complex query is, the higher is the impact of generalization search. Obviously, one cannot expect any improvement by a typical search query of 2-3 keywords. For 5-7 keywords one observes some accuracy decline, which is reversed when a query is a whole sentence or more, up to three sentences. Overall improvement of search relevancy (as defined above) for 50 queries (above 3 keywords) is 4.0%, which is noticeable

for a user (although not a reason to switch away from a favorite search engine).

Type of search query	Relevancy of Yahoo search, %, averaging over 10	Relevancy of re-sorting by generalization, %, averaging over 10	improvement of relevancy, %
3-4 word phrases	77	77	100.0%
5-7 word phrases	79	78	98.7%
8-10 word single sentences	77	80	103.9%
2 sentences, >8 words total	77	83	107.8%
3sentences,>12 words total	75	82	109.3%

Table1: Evaluation of improvement of search accuracy using syntactic generalization.

Results and conclusions

In this study we defined sentence generalization and generalization diagrams which can be constructed automatically from syntactic parse trees and support semantic classification task. Similarity measure between syntactic parse trees is developed as a generalization operation on the lists of sub-trees of these trees. This operation is defined as extension of anti-unification of logic formulas towards such language structures as syntactic parse trees.

The diagrams are representation of mapping between the level of syntactic generalization and semantic one (unti-unification of logic forms). Generalization diagrams are intended to be more accurate than conceptual graphs for individual sentences, because only syntactic commonalities are represented at semantic level. Use of commonalities for reliable construction of meaning representation follows along the line of the canons of induction. It states that if two or more instances of a phenomenon under investigation have only one circumstance in common, then this circumstance is the cause or effect of the given phenomenon. Generalization diagrams are then used for semantic classification, where classes are characterized by this phenomenon. Hence proposed generalization diagrams are expected to be well suited for text learning tasks. In this study we demonstrated that use of generalization diagrams indeed improves the search relevancy.

In this study we showed how mapping the syntactic generalization into semantic generalization allowed to treat text relevance in a systematic way. Having evaluated the relevance in industrial setting (allvoices.com and zvents.com) by providing content to more than 20 million users monthly, we conclude the proposed approach is robust and scalable to a wide majority of text analysis and search applications

References

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

Cardie, C., Mooney R.J. Machine Learning and Natural Language. Machine Learning 1(5) 1999.

Ravichandran, D and Hovy E. 2002. Learning surface text patterns for a Question Answering system. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002), Philadelphia, PA.

Polovina S. and John Heaton, "An Introduction to Conceptual Graphs," AI Expert, pp. 36-43, 1992.

Sowa JF, Information Processing in Mind and Machine. Reading, MA: Addison-Wesley Publ., 1984.

Dzikovska, M., M. Swift, Allen, J., William de Beaumont, W. (2005). Generic parsing for multi-domain semantic interpretation. International Workshop on Parsing Technologies (Iwpt05), Vancouver BC.

Banko, Michael J. Cafarella, Stephen Soderland,Matt Broadhead, and Oren Etzioni. 2007 Open information extraction from the web. In *Proceedingsof the Twentieth International Joint Conference on Artificial Intelligence*, pages 2670–2676, Hyderabad, India. AAAI Press.

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

B.Galitsky, G. Dobrocsi, J.L. de la Rosa, S.O. Kuznetsov: From Generalization of Syntactic Parse Trees to Conceptual Graphs, in M. Croitoru, S. Ferré, D. Lukose (Eds.): Conceptual Structures: From Information to Intelligence, 18th International Conference on Conceptual Structures, ICCS 2010, Lecture Notes in Artificial Intelligence, vol. 6208, pp. 185-190.

Plotkin. GD A note on inductive generalization. In B. Meltzer and D. Michie, editors, *Machine Intelligence*, volume 5, pages 153-163. Elsevier North-Holland, New York, 1970.

Robinson JA. A machine-oriented logic based on the resolution principle. *Journal of the Association for Computing Machinery*, 12:23-41, 1965