Under the Hood: The natural language interface of Graph Search

April 29, 2013 at 10:08am

Xiao Li is an engineering manager on the natural language team for Graph Search and Maxime Boucher is a research scientist for Graph Search.

The Graph Search engine is built upon highly structured data in the form of a graph, representing hundreds of types of nodes and thousands of types of edges. Users, Pages, places, photos and posts are all nodes in the graph, each with structured information of its own nature. For example, users have gender information, places have addresses, and photos have posting dates. Moreover, the nodes are connected to each other in various ways. A user can like a Page, study at a school, live in a city, be in a relationship with another user, check in at a place, and comment on a photo. A photo, in turn, can be tagged with a user, and be taken at a place. It is the richness of the data that defines the nature of Graph Search; the system needs to be designed toward understanding the user intent precisely and serving structured objects.

Given this large variety of nodes and edges, building a search engine to let people search over Facebook's graph has proven to be a great challenge. The Graph Search team iterated over possible query interfaces at the early stage of this project. There was consensus among the team that a keyword-based system would not be the best choice because of the fact that keywords, which usually consist of nouns or proper nouns, can be nebulous in their intent. For example, "friends Facebook" can mean "friends on Facebook," "friends who work at Facebook Inc.," or "friends who like Facebook the Page." Keywords, in general, are good for matching objects in the graph but not for matching connections between the objects. A query built on keywords would fail in cases where a user needs to precisely express intent in terms of both nodes and edges in the graph. The team also toyed with the idea of form-filling augmented by drop-down filters. However, because of all the possible options you could search for in Facebook's data, this would easily lead to an interface of hundreds of filters.

In mid-2011, the team converged around the idea of building a natural language interface for Graph Search, which we believe to be the most natural and efficient way of querying the data in Facebook's graph. You can find "TV shows liked by people who study linguistics" by issuing this query verbatim and, for the entertainment value, compare the results with "TV shows liked by people who study computer science." Our system is built to be robust to many varied inputs, such as grammatically incorrect user queries, and can also recognize traditional keyword searches. Our query suggestions are always constructed in natural language, expressing the precise intention interpreted by our system. This means you know in advance whether the system has correctly understood your intent before selecting any suggestion and executing a search. The system also suggests options for completing your search as you type in to the typeahead, demonstrating what kinds of queries it can understand.

The components of the architecture of our natural language interface are:

1. Entity recognition and resolution, i.e., finding possible entities and their categories in an input query and resolving them to database entries.
2. Lexical analysis, i.e., analyzing the morphological, syntactical and semantic information of the words/phrases in the input query.
3. Semantic parsing, i.e., finding the top "N" interpretations of an input query given a grammar expressing what one can potentially search for using Graph Search.

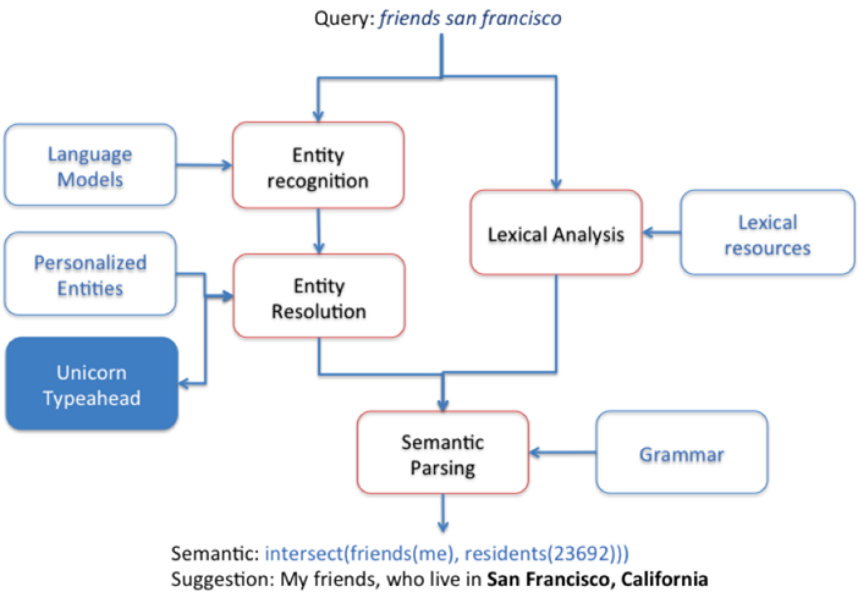
Facebook Engineering

Notes by Facebook Engineering

All Notes

Get Notes via RSS

Embed Post

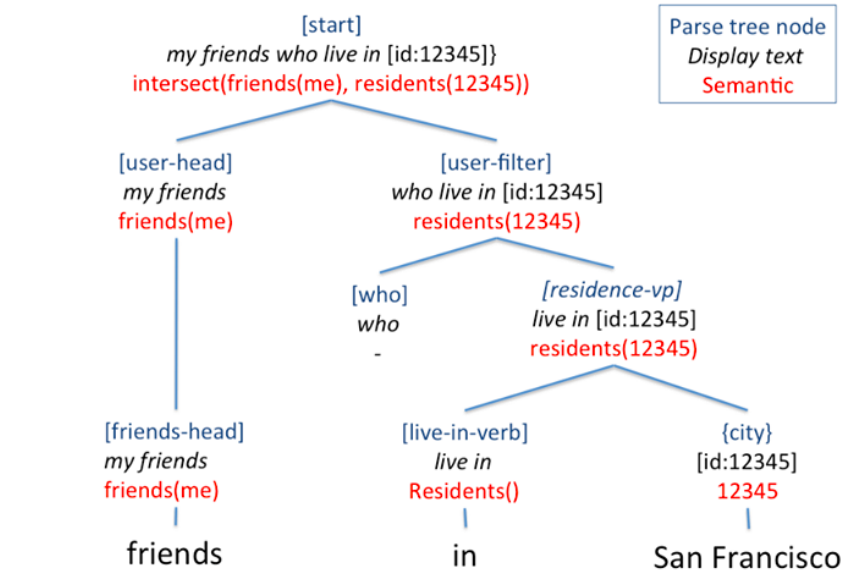


Grammar

Structure: We use a weighted context free grammar (WCFG) to represent the Graph Search query language, defining what queries can be understood by Graph Search. In loose terms, the grammar consists of a set of production rules that generate more specific expressions from abstract symbols:

[start] => [users]	$1
[users] => my friend	friends(me)
[users] => friends of [users]	friends($1)
[users] => {user}	$1
[start] => [photos]	$1
[photos] => photos of [users]	photos($1)

The symbol [start] is the root of a parse tree. The left-hand-side of the rule is a non-terminal symbol, producing the right-hand-side, which consists of either non-terminal or terminal symbols. In Graph Search, a terminal symbol can be an entity, e.g., {user}, {city}, {employer}, {group}; it can also be a word/phrase, e.g., friends, live in, work at, members, etc. A parse tree is produced by starting from [start] and iteratively expanding the production rules until it reaches terminal symbols.



The parse tree, semantic and entity ID used in the above example are for illustration only; they do not represent real information used in Graph Search Beta

Semantic: Each production rule has a semantic function, and each parse tree, therefore, is associated with a semantic tree. A semantic function can take arguments such as an entity ID in the rule, if available, and modifiers of an entity category, and semantic functions are combined to form semantic trees. For example, the parse tree that generates "My friends who live in {city}" has a semantic intersect(friends(me), residents(12345))). Such semantics can be transformed to the Unicorn language (link to: https://www.facebook.com/notes/facebook-engineering/under-the-hood-building-out-the-infrastructure-for-graph-search/10151347573598920) and then executed against search indexes.

Parameterization: The grammar has a cost structure in order to produce relative rankings of parse trees. Our grammar currently has three large categories of costs:

1. Rule costs (query-independent): This set of costs represents prior information in the rules themselves. A rule cost is determined by both the semantic and the display text associated with a rule.
2. Entity costs (query and searcher-dependent): These are costs of matching entities in terminal rules, which depends on the outputs of entity detection as well as entity resolution.
3. Matching costs (query-dependent): This category of costs is for matching lexical tokens in terminal rules. This category includes insertion costs, deletion costs, substitution costs and transposition costs. The table below gives a summary of what these operations represent.

	Original input query	Suggestion
Insertion	Friends San Francisco	My friends who live in San Francisco
Deletion	Which friends live in San Francisco	My friends who live in San Francisco
Substitution	My best friends who live in San Francisco	My closest friends who live in San Francisco
Transposition	San Francisco friends	My friends who live in San Francisco

Next, we take a closer look at how an input query is matched against entities and lexical tokens in a terminal rule.

Entity detection and resolution

As mentioned earlier, the terminal rules of the grammar consist of entities as well as words and phrases. To detect entities, we built a detector that can identify query segments that are

likely to be entities and classify those segments into entity categories. For example,

- “people who live in san francisco”

The segment “san francisco” is likely to be an entity of the {city} category.

In Graph Search, we have 20+ entity categories, including {user}, {group}, {application}, {city}, {college}, etc. At entity detection time, we allow multiple query segments, including overlapping ones, to be detected as potential entities, and allow multiple entity categories to be assigned to each query segment. This process provides important signals for semantic parsing.

The entity detector is constructed on the basis of n-gram based language models. Such models contain conditional probabilities of a word given the past n-1 words as well as smoothing parameters, providing a principled way of estimating how likely a word sequence is generated by a data source. In the context of Graph Search, we built two types of language models:

- A set of entity language models, each represented by n-gram statistics for an entity category. For example, the bigrams san+francisco and new+york both have high probabilities in the {city} language model, but low probabilities in the {user} language model.
- A grammar language model, represented by n-gram statistics of the Graph Search query language. For example, live+in+{city} is a prominent trigram in the grammar language model.

Given these two types of language models, one can perform inference on a given input query to estimate the probability of any query segment belonging to any entity category, i.e.,

p(Class(Qi:j)= K | Q1:N),for all {i, j, k}

For query segments that are detected as entities with high confidence, we send them to the Unicorn typeahead system for entity resolution, i.e., retrieving and ranking database entries given the text form of that entity. The Unicorn typeahead system ranks entities based on signals such as static rank, text proximity, social proximity, and geographical proximity, among many others. These systems have been described in a previous blog post here: https://www.facebook.com/notes/facebook-engineering/under-the-hood-indexing-and-ranking-in-graph-search/10151361720763920.

Lexical analysis

The grammar that powers Graph Search was developed to let users query for any sets of results. Our team realized very early on that to be useful, the grammar should also allow users to express their intent in many various ways. For example, a user can search for photos of his or her friends by typing:

- “photos of my friends”
- “friend photos”
- “photos with my friends”
- “pictures of my friends”
- “photos of facebook friends”;

A user should be able to find people who are interested in surfing by:

- “people who like surfing”
- “people who surf”
- “surfers”

Moreover, we should allow users to issues queries that arenot necessarily grammatically correct, e.g.,

- “people who works at facebook”
- “photo of my friends”

The challenge for the team was to make sure that any reasonable user input produces plausible suggestions using Graph Search. To achieve that goal, the team leveraged a number of linguistic resources for conducting lexical analysis on an input query before matching it against terminal rules in the grammar.

Synonyms: The team gathered long lists of synonyms that we felt could be used interchangeably. Using synonyms, one can search for “besties from my hood” and get the same results as if you had searched for “my friends from my hometown”. Note that matching synonyms comes with a cost, i.e., the substitution cost described in grammar parameterization. In addition, there are cases that go beyond word level synonymization; an input query form can be reformulated to another form as a paraphrase, e.g.,

- “where to eat sushi” -> *Sushi restaurants*

Fillers: Our grammar only covers a small subspace of what a user can potentially search for. There are queries that cannot be precisely answered by our system at this time but can be approximated by certain forms generated from the grammar. For example,

- “all my friends photos” -> *My friends’ photos*
- “young men from my hometown” -> *Men from my hometown*

In order for our grammar to focus on the most important parts of what a user types, the team built a list of words that can be optionalized in certain contexts: “all” can be ignored when it appears before a head noun as in “all photos”, but shouldn’t be ignored in other contexts such as “friends of all” (which could be auto completed to “friends of Allen” and thus shouldn’t be optionalized). Various corpora such as the Penn Treebank were used to gather words that can be optionalized in certain sub-trees of the grammar.

Related forms: Some entities in the Facebook graph correspond to general concepts that can be described with different forms. For example, on a Facebook profile it is possible to list “surfing” as an interest. However, if one searches for “people who surf”, or “surfers”, one could reasonably expect to find that person, even though the page liked by that person is “surfing”. Our team used WordNet to extract related word forms to let users search for people with similar interests in very simple queries “surfers in Los Angeles” or “quilters nearby”.

Inflections: We also obtained inflected forms for the nouns and verbs in our terminal rules. Some inflections do not change semantics, e.g., “photo” vs. “photos”, while others (such as tenses) can carry different semantics, e.g., “people who work at facebook” vs. “people who worked at facebook”. Moreover, some inflections require agreements with other parts of a sentence. Our system only returns parse trees in which all word forms are in accordance with all agreement rules. As result, the display text for any input query is always grammatically correct, e.g.:

“people who works at facebook” -> *People who work at Facebook*
“photo of my friends” -> *Photos of my friends*

Parsing
The input query, augmented by entity information and lexical analysis, is then fed into a semantic parser, which outputs the “K” best parse trees as well as their semantics and display texts. Parsing is performed in three steps.

Terminal rule matching: Find all terminal rules from the grammar that match the input query. During this process, we also obtain the information about:

1. The starting and ending positions of the query, (*i, j*), against which each rule matches.
2. The cost associated with each rule and query segment pair, (*Rk, Qi,j*). The cost is computed based on editing costs described in our grammar parameterization, as well as rule costs themselves.

Search: The parsing step aims at constructing a parse tree, and hence its semantic tree, from a subset of matched (*Rk, Qi,j*). This subset has to contain a sequence of non-overlapping, consecutive token ranges, (*i, j*), that spans the entire input query. The parse tree constructed has to reach the [start] symbol.

In Graph Search, we use a variation of the N-shortest path algorithm, an extension of Dijkstra’s algorithm, to solve the problem of finding the top K best parse trees. Our biggest challenge was to find several heuristics that allow us to speed up the computation of the top K grammar suggestions, thereby providing a real-time experience to our users.

Semantic scoring:

A naïve, context-free grammar would allow the production of a wide range of sentences, some of which can be syntactically correct but not semantically meaningful. For example:

- Non-friends who are my friends
- Females who live in San Francisco and are males

Both sentences would return empty sets of results because they each carry contradictory semantics. It is therefore critical for our parser to be able to understand semantics with opposite meanings in order to return plausible suggestions to users.

We also need to prevent Graph Search from presenting multiple suggestions that have the same meaning, e.g.:

- My friends
- People who are my friends
- People I am friends with

It would be bad user experience to see these suggestions appear together, as they are essentially different ways of expressing the same intent.

To prevent the parser from producing suggestions that are semantically implausible, or producing multiple suggestions with duplicate semantics, we built constraints into the grammar, and modified our search algorithm to comply with those constraints. In particular, we used a semantic scoring mechanism that demotes or rejects undesirable suggestions during the search process of finding top K parse trees.

Fall back: Finally, for queries completely out of the scope of our grammar, we built a mechanism to detect them and fall back to entity suggestions or web search suggestions. The detector uses a number of features extracted from the input query and the parse trees to decide whether the user intent is covered by the scope of our grammar.

Going Forward

At the time we launched Graph Search, there was little real user data that could be used to optimize our system, and a good number of the components here were designed based on intuition and tuned based on a limited set of data samples. We are excited to see how Graph Search performs now it has begun rolling out, and to use that data to improve how search suggestions are generated. Closing the feedback loop for Graph Search will be a big step toward a data-driven system optimized for user engagement and satisfaction.

There are many exciting milestones ahead of us. Making Graph Search available to mobile and international users will give all users equal opportunities to enjoy the power of Graph Search. The grammar coverage can be expanded drastically if we inject semantic knowledge from the outside of the Facebook graph, and connect it with the Facebook world. As a simple example, we would be able to serve answers to “Steven Spielberg movies liked by my friends” by finding connections between Steven Spielberg and movies in our graph.

Graph Search is one important pillar of Facebook in our mission to make the world more open and connected. We are only 1% complete, and we are looking forward to the rest of the journey!


Kalle Kala I need Help !!! Hello, This is not the first nor the last time I have to write to you. How long will this problem that i healed my account FB-s? I had to create a new account but only in extreme cases. I want my old account back, however, there is a th... See More
May 23, 2013 at 2:41am


Nikhil Soni Awesome pic..plz like dis photo in the link
Nd share it...
http://m.facebook.com/photo.php?fbid=10151330448791116&id=38069396115&set=a.10151318240471116.1073741826.38069396115&refid=56
May 19, 2013 at 12:03pm


i m nt geettng any code login approval.....i cldnt acess my accnt Plz help me
May 17, 2013 at 11:14am


D Smith how about giving devs deeper access to this via the API so we can build apps around the query results
May 8, 2013 at 7:19am


문원기 Good information!
May 8, 2013 at 1:28am


Brady Gibson Gibmatic Pro Make a short verson pliz
May 6, 2013 at 4:54am


นพรัตน์ พเนจร THA
May 5, 2013 at 10:00pm


Josshua Calixto' nice roundup of your process. It's great to see natural language processing techniques in a real workflow.
□□□□□□□□
May 5, 2013 at 1:58pm


Filip Haglund so, the grammar nazis found a typo on the first row, but no mention of "san fransico" which is even underlined? lazy nazis, maybe?

great text!
1 · April 29, 2013 at 10:31am


Hasan Kamal Tahir okay..
1 · April 29, 2013 at 10:08am


Shubham Nishad hmm, so long article. is there any short version of it?
2 · April 29, 2013 at 10:12am


Brian Swichkow Why is this no longer working? When you did the update you removed the functionality to search things like "my friends who live near __". I travel constantly and that was one of my favorite ways to reconnect.
December 25, 2014 at 2:13pm

Keshav Anand Great post team. Machine learning would be much better if it had ability to infer rather than just landing up on some node access based result.
May 20, 2014 at 11:41am


Martin Claxton IBM predicted this event over 15 years ago. I was a WebSphere student at the time. No name or face on the company. Just the concept of products finding you. I don't think many people appreciate what has really happened here. This is awesome from any p... See More
April 7, 2014 at 6:22am


Renaud Richardet how many people are regularily using graph search features?
February 4, 2014 at 2:41am


Prateek Gupta cool stuff..
July 22, 2013 at 10:23pm

Morena Saai ur line has being improved we as colleques facebook we appriate ur news.
July 3, 2013 at 8:38am


Carmen Cornejo R Interesting note, there will be a short version?
June 30, 2013 at 11:47am

Kouadio Békanti Sylvain Anthelme j'adore l'ordinateur
June 19, 2013 at 2:20am


Faye Xu How can be the processing run in a user accept time? like to see
May 23, 2013 at 4:54am


Kalle Kala I can not find a place to turn to your request??
May 23, 2013 at 2:43am


Doug McMillan Can someone please tell me how to LOG OUT of this NEW FB!
These people seem to be UN-Content with their Product & change formats like peeps change underwear! I shut down profile cuz if this & just opened it back up! May be Better for You Unsatisfied so... See More
May 6, 2013 at 6:19am


Ana Maria Popescu Nice write-up - very informative. If you published a longer white paper/industry paper, I think it'd be of interest to a lot of people.
May 2, 2013 at 10:28pm


Roy van Tilborg hallo mag ik eens vragen waarom we niet een knop hebben op facebook de knop vindt ik niet leuk ,want niet alles is leuk wat we op facebook zetten.
See Translation
May 2, 2013 at 2:18pm


Prerak Pradhan did do this on top of the DBpedia engine using wordnet, NER and a dependency parser

May 2, 2013 at 12:45am

Pcm Smc New our ad has been disapproved by FB even with >50 revisions. we've contacted FB HK to help but no feedback after a week. how can i get any support fr FB?
May 1, 2013 at 7:43pm

Kaufman Ng this article could benefit from semantic parsing, a few typos
April 30, 2013 at 8:54pm

Andres Hohendahl I am working on a similar approach, but defined a new language to do so, and now things are more easy.
April 30, 2013 at 7:16pm

Wang Dong glad to know parsing tree is used here.
April 30, 2013 at 3:52pm

Santhosh Kumar Graph search – I started playing with this and see that this phenomenal... How intuitive would this for mobile users ?
April 30, 2013 at 12:51pm

Yokesh A Graph Search Just a start 1% ? hah i hope i live till it reach least 80% of it''s potential
April 30, 2013 at 4:06am

Deva Reddy Jyothi We implemented different approach with deeper semantics in our Semantics social platform SOCOTO for graph search with natural interface that can connect to out side world also. Increases scope intent and relevance http://www.youtube.com/watch?v=Zvjj0q0n9LU and http://www.evamtec.com
1 · April 30, 2013 at 3:44am

Victor Rodriguez Great!
April 29, 2013 at 8:41pm

Steven Zuniga some things just get better with use. The feeling of wearing something in, than wearing out, comes to mind.
April 29, 2013 at 6:10pm

Patrick Gibson That link just opens a blurry thumbnail picture of some chart from the Facebook app on the phone. Known issue?
April 29, 2013 at 3:33pm

Abhishek Sagar awesome write up, loved it
April 29, 2013 at 2:38pm

Tariq Wali It's just BS you could't get 'data in` right
April 29, 2013 at 1:17pm

Michał Ty I still don't have graph search, even though I've applied for it at or close to the very moment it was made available
April 29, 2013 at 12:03pm

Kiyas KA Gd 9t
April 29, 2013 at 10:36am

Faizan Chaudhry gud nit
April 29, 2013 at 10:16am

View more comments