Presented at DARPA Human Language Technology Conference, March 2001, San Diego, CA.

Toward Semantics-Based Answer Pinpointing

Eduard Hovy, Laurie Gerber, Ulf Hermjakob, Chin-Yew Lin, Deepak Ravichandran

Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, CA 90292-6695
Tel: 310-448-8731
Fax: 310-823-6714
Email: {hovy,gerber,ulf,cyl,deepak}@isi.edu

1. Introduction

Several research projects have recently investigated the problem of automatically answering simple questions that have brief phrasal answers ('factoids'), by identifying and extracting the answer from a large collection of text.

The systems built in these projects exhibit a fairly standard structure: they create a query from the user's question, perform IR with the query to locate (segments of) documents likely to contain an answer, and then pinpoint the most likely answer passage within the candidate documents. The most common difference lies in the pinpointing. Many projects employ a window-based word scoring method that rewards desirable words in the window. They move the window across the candidate answers texts/segments and return the window at the position giving the highest total score. A word is desirable if it is a content word and it is either contained in the question, or is a variant of a word contained in the question, or if it matches the words of the expected answer. Many variations of this method are possible—of the scores, of the treatment of multi-word phrases and gaps between desirable words, of the range of variations allowed, and of the computation of the expected answer words.

Although it works to some degree (giving results of up to 30% in independent evaluations), the window-based method has several quite serious limitations:

- it cannot pinpoint answer boundaries precisely (e.g., an exact name or noun phrase),
- it relies solely on information at the word level, and hence cannot recognize information of the desired type (such as Person or Location),
- it cannot locate and compose parts of answers that are distributed over areas wider than the window.

Window-based pinpointing is therefore not satisfactory in the long run, even for factoid QA. In this paper we describe work in our Webclopedia project on semantics-based answer pinpointing. Initially, though, recognizing the simplicity and power of the window-based technique for getting started, we implemented a version of it as fallback method. We then implemented two more sophisticated methods: syntactic-semantic question analysis and QA pattern matching. This involves classification of QA types to facilitate recognition of desired answer types, a robust syntactic-semantic parser to analyze the question and candidate answers, and a matcher that combines word- and parse-tree-level information to identify answer passages more precisely. We expect that the two methods will really show their power when more complex non-factoid answers are sought. In this paper we describe how well the three methods did relative to each other. Section 2 outlines the Webclopedia system. Sections 3, 4, and 5 describe the semantics-based components: a QA Typology, question and answer parsing, and matching. Finally, we outline current work on automatically learning QA patterns using the Noisy Channel Model.

2. Webclopedia

Webclopedia's architecture (Figure 1) follows the pattern outlined above:

Question parsing: Using BBN's IdentiFinder (Bikel et al., 1999), the CONTEX parser (Section 4) produces a syntactic-semantic analysis of the question and determines the QA type (Section 3).

Query formation: Single- and multi-word units (content words) are extracted from the analysis, and WordNet synsets are used for query expansion. A Boolean query is formed. Details in (Hovy et al., 2000).

IR: The IR engine MG (Witten et al., 1994) returns the top-ranked 1000 documents.

Segmentation: To decrease the amount of text to be processed, the documents are broken into semantically coherent segments. Two text segmenter—TexTiling (Hearst, 1994), C99 (Choi, 2000)—were tried; the first is used. See (Hovy et al., 2000).

Ranking segments: For each segment, each sentence is scored using a formula that rewards word and phrase overlap with the question and its expanded query words. Segments are ranked. See (Hovy et al., 2000)

Parsing segments: CONTEX parses each sentence of the top-ranked 100 segments (Section 4).

Pinpointing: For each sentence, three steps of matching are performed (Section 5); two compare the analyses of the question and the sentence; the third uses the window method to compute a goodness score.

Ranking of answers: The candidate answers' scores are compared and the winner(s) are output.

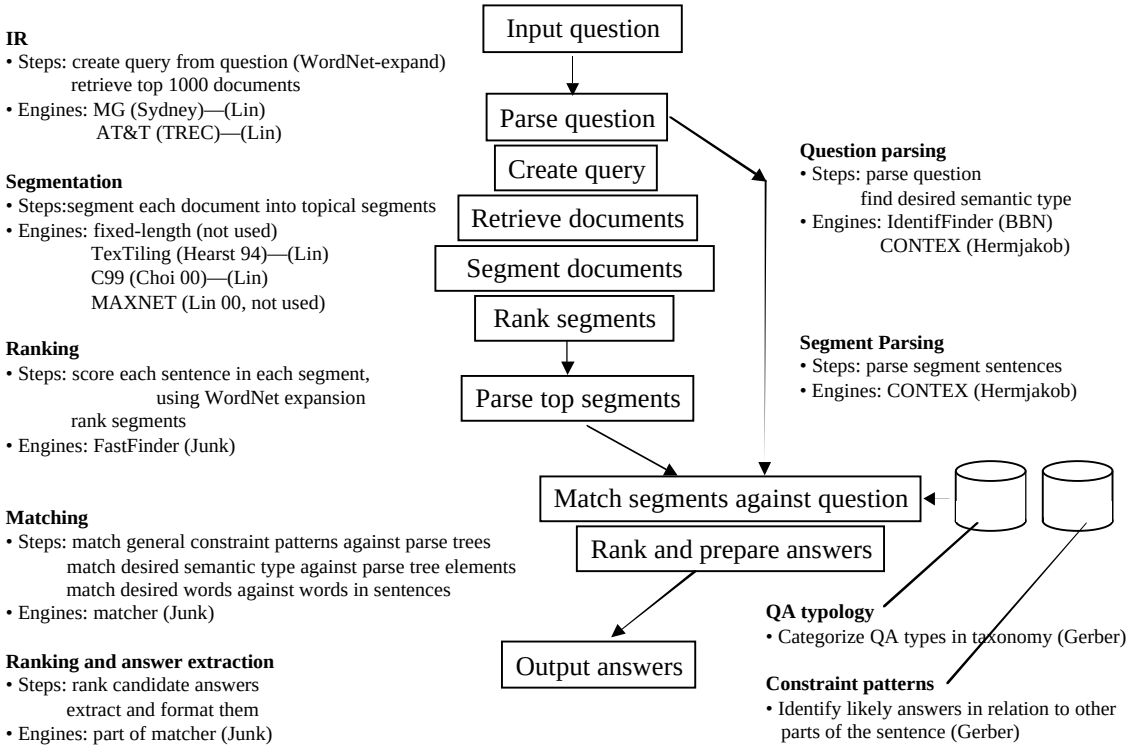

Figure 1. Webclopedia architecture.

3. The QA Typology

In order to perform pinpointing deeper than the word level, the system has to produce a representation of what the user is asking. Some previous work in automated question answering has categorized questions by question word or by a mixture of question word and the semantic class of the answer (Srihari and Li, 2000; Moldovan et al., 2000). To ensure full coverage of all forms of simple question and answer, and to be able to factor in deviations and special requirements, we are developing a QA Typology.

We motivate the Typology (a taxonomy of QA types) as follows.

There are many ways to ask the same thing: *What is the age of the Queen of Holland? How old is the Netherlands' queen? How long has the ruler of Holland been alive?* Likewise, there are many ways of delivering the same answer: *about 60; 63 years old; since January 1938*. Such variations form a sort of semantic equivalence class of both questions and answers. Since the user may employ any version of his or her question, and the source documents may contain any version(s) of the answer, an efficient system should group together equivalent question types and answer types. Any specific question can then be indexed into its type, from which all equivalent forms of the answer can be ascertained. These QA equivalence types can help with both query expansion and answer pinpointing.

However, the equivalence is fuzzy; even slight variations introduce exceptions: *who invented the gas laser?* can be answered by both *Ali Javan* and *a scientist at MIT*, while *what is the name of the person who invented the gas laser?* requires the former only. This inexactness suggests that the QA types be organized in an inheritance hierarchy, allowing the answer requirements satisfying more general questions to be overridden by more specific ones 'lower down'.

These considerations help structure the Webclopedia QA Typology. Instead of focusing on question word or semantic type of the answer, our classes attempt to represent the user's intention, including for example the classes Why-Famous (for *Who was Christopher Columbus?* but not *Who discovered America?*, which is the QA type Proper-Person) and Abbreviation-Expansion (for *What does HLT stand for?*). In addition, the QA Typology becomes increasingly specific as one moves from the root downward.

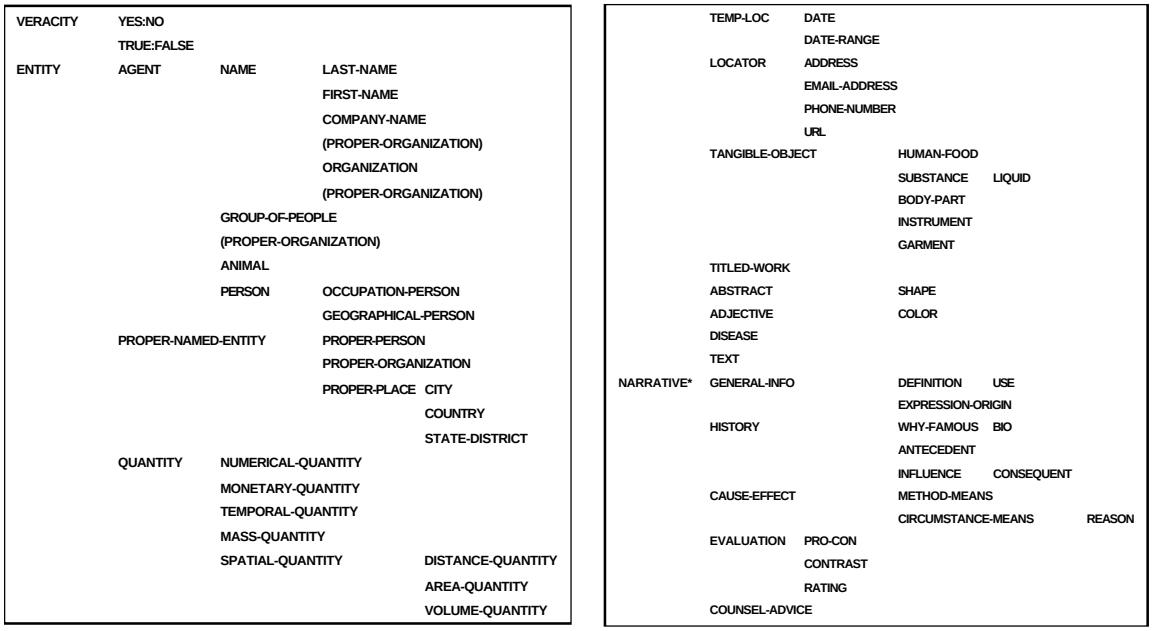

Figure 2. Portion of Webclopedia QA Typology.

To create the QA Typology, we analyzed 17,384 questions and their answers (downloaded from answers.com); see (Gerber, in prep.). The Typology (Figure 2) contains 79 nodes, whose leaf nodes capture QA variations that can in many cases be further differentiated. Each Typology node has been annotated with examples and typical patterns of expression of both Question and Answer, using a simple template notation that expressed configurations of words and parse tree annotations (Figure 3). Question pattern information (specifically, the semantic type of the answer required, which we call a Qtarget) is used by the CONTEX parser (Section 4) when analyzing the question, enabling it to output its guess(s) for the QA type. Answer pattern information is used by the Matcher (Section 5) to pinpoint likely answer(s) in the parse trees of candidate answer sentences.

Question examples	**Question templates**
Who was Johnny Mathis' high school track coach?	who be <entity>'s <role>
Who was Lincoln's Secretary of State?	
Who was President of Turkmenistan in 1994?	who be <role> of <entity>

Who is the composer of Eugene Onegin?
Who is the chairman of GE?

Actual answers	**Answer templates**
Lou Vasquez, track coach of…and Johnny Mathis	<person>, <role> of <entity>
Signed Saparmurad Turkmenbachy [Niyazov], president of Turkmenistan	<person> <role-title*> of <entity>
...Turkmenistan's President Saparmurad Niyazov…	<entity>'s <role> <person>
...in Tchaikovsky's Eugene Onegin...	<person>'s <entity>
Mr. Jack Welch, GE chairman...	<role-title> <person> ... <entity> <role>
...Chairman John Welch said ...GE's	<subject>|<psv object> of related role-verb

Figure 3. Portion of QA Typology node annotations for Proper-Person.

At the time of the TREC-9 Q&A evaluation, we had produced approx. 500 patterns by simply cross-combining approx. 20 Question patterns with approx. 25 Answer patterns. To our disappointment (see Section 6), these patterns were both too specific and too few to identify answers frequently—when they applied, they were very accurate, but they applied too seldom. We therefore started work on automatically learning QA patterns in parse trees (Section 7). On the other hand, the semantic class of the answer (the Qtarget) is used to good effect (Sections 4 and 6).

4. Parsing

CONTEX is a deterministic machine-learning based grammar learner/parser that was originally built for MT (Hermjakob, 1997). For English, parses of unseen sentences measured 87.6% labeled precision and 88.4% labeled recall, trained on 2048 sentences from the Penn Treebank. Over the past few years it has been extended to Japanese and Korean (Hermjakob, 2000).

4.1 Parsing Questions

Accuracy is particularly important for question parsing, because for only one question there may be several answers in a large document collection. In particular, it is important to identify as specific a Qtarget as possible. But grammar rules for declarative sentences do not apply well to questions, which although typically shorter than declaratives, exhibit markedly different word order, preposition stranding ("What university was Woodrow Wilson President of?"), etc.

Unfortunately for CONTEX, questions to train on were not initially easily available; the Wall Street Journal sentences contain a few questions, often from quotes, but not enough and not representative enough to result in an acceptable level of question parse accuracy. By collecting and treebanking, however, we increased the number of questions in the training data from 250 (for our TREC-9 evaluation version of Webclopedia) to 400 on Oct 16 to 975 on Dec 9. The effect is shown in Table 1. In the first test run ("[trained] without [additional questions]"), CONTEX was trained mostly on declarative sentences (2000 Wall Street Journal sentences, namely the enriched Penn Treebank, plus a few other non-question sentences such as imperatives and short phrases). In later runs ("[trained] with [add. questions]"), the system was trained on the same examples plus a subset of the 1153 questions we have treebanked at ISI (38 question from the pre-TREC-8 test set, all 200 TREC-8 and 693 TREC-9 questions, and 222 non-TREC questions).

			Labeled	Labeled	Tagging	Crossing
	Precision	Recall	Precision	Recall	Accuracy	Brackets
Without, Oct 16	90.74%	90.72%	84.62%	83.48%	94.95%	0.6
With, Oct 16	94.19%	94.86%	91.63%	91.91%	98.00%	0.48
With, Dec 9	97.33%	97.13%	95.40%	95.13%	98.64%	0.19

Table 1. Improvement in parsing of questions.

The TREC-8 and TREC-9 questions were divided into 5 subsets, used in a five-fold cross validation test in which the system was trained on all but the test questions, and then evaluated on the test questions.

Reasons for the improvement include (1) significantly more training data; (2) a few additional features, some more treebank cleaning, a bit more background knowledge etc.; and (3) the 251 test questions on Oct. 16 were probably a little bit harder on average, because a few of the TREC-9 questions initially treebanked (and included in the October figures) were selected for early treebanking because they represented particular challenges, hurting subsequent Qtarget processing.

4.2 Parsing Potential Answers

The semantic type ontology in CONTEX was extended to include 115 Qtarget types, plus some combined types; more details in (Hermjakob, 2001). Beside the Qtargets that refer to concepts in CONTEX's concept ontology (see first example in Figure 2), Qtargets can also refer to part of speech labels (first example), to constituent roles or slots of parse trees (second and third examples), and to more abstract nodes in the QA Typology (later examples). For questions with the Qtargets Q-WHY-FAMOUS, Q-WHY-FAMOUS-PERSON, Q-SYNONYM, and others, the parser also provides Qargs—information helpful for matching (final examples).

Semantic ontology types (I-EN-CITY) **and part of speech labels** (S-PROPER-NAME):

What is the capital of Uganda?
QTARGET: (((I-EN-CITY S-PROPER-NAME)) ((EQ I-EN-PROPER-PLACE)))

Parse tree roles:

Why can't ostriches fly?	QTARGET: (((ROLE REASON)))
Name a film in which Jude Law acted.	QTARGET: (((SLOT TITLE-P TRUE)))

QA Typology nodes:

What are the Black Hills known for?	Q-WHY-FAMOUS
Who was Whitcomb Judson?	Q-WHY-FAMOUS-PERSON
What is Occam's Razor?	Q-DEFINITION
What is another name for nearsightedness?	Q-SYNONYM
Should you exercise when you're sick?	Q-YES-NO-QUESTION
True or false: Chaucer was an actual person.	Q-TRUE-FALSE-QUESTION

Qargs for additional information:

Who was Betsy Ross? QTARGET: (((Q-WHY-FAMOUS-PERSON))) QARGS: (("Betsy Ross"))
How is "Pacific Bell" abbreviated? QTARGET: (((Q-ABBREVIATION))) QARGS: (("Pacific Bell"))
What are geckos? QTARGET: (((Q-DEFINITION))) QARGS: (("geckos" "gecko") ("animal"))

Figure 2. QA-related information in the parse tree of the question.

These Qtargets are determined during parsing using 276 hand-written rules. Still, for approx. 10% of the TREC-8&9 questions there is no easily determinable Qtarget ("What does the Peugeot company manufacture?"; "What is caliente in English?"). Strategies for dealing with this are under investigation. More details appear in (Hermjakob, 2001). The current accuracy of the parser on candidate answer sentences is shown in Table 2.

# Penn Treebank sentences	# Question sentences added	Labeled Precision	Labeled Recall	Tagging Accuracy	Crossing brackets (/ sent)	Qtarget accuracy (strict)	Qtarget accuracy (lenient)
2000	0	83.47%	82.49%	94.65%	0.34	63.00%	65.50%
3000	0	84.74%	84.16%	94.51%	0.35	65.30%	67.40%
2000	38	91.20%	89.37%	97.63%	0.26	85.90%	87.20%
3000	38	91.52%	90.09%	97.29%	0.26	86.40%	87.80%
2000	975	95.71%	95.45%	98.83%	0.17	96.10%	97.30%

Table 2. Answer parse tree and Qtarget accuracies.

5. Answer Matching

The Matcher performs three independent matches, in order:

- match QA patterns in the parse tree,
- match Qtargets and Qwords in the parse tree,
- match over the answer text using a word window.

Details appear in (Hovy et al., 2000).

6. Results

We entered the TREC-9 short form QA track, and received an overall Mean Reciprocal Rank score of 0.318, which put Webclopedia in essentially tied second place with two others. (The best system far outperformed those in second place.)

In order to determine the relative performance of the modules, we counted how many correct answers their output contained, working on our training corpus. Table 3 shows the evolution of the system over a sample one-month period, reflecting the amount of work put into different modules. The modules QA pattern, Qtarget, Qword, and Window were all run in parallel from the same Ranker output.

Date	Number Qs	IR hits	Ranker hits	QA pattern	Qtgt match	Qword fallback	Window fallback	Total
2-Jul	52	1.00	0.61	0.12	0.49	0.15	0.19	0.62
8-Jul	38	0.89	0.40	0.28	0.40	0.12	n/a	0.53
13-Jul	52	1.00	0.61	0.04	0.48	0.15	0.22	0.53
3-Aug	55	n/a	n/a	0.04	0.32	0.15	0.19	0.41

Table 3. Relative performance of Webclopedia modules on training corpus.

The same pattern, albeit with lower scores, occurred in the TREC test (Table 4). The QA patterns made only a small contribution, the Qtarget made by far the largest contribution, and, interestingly, the word-level window match lay somewhere in between.

IR hits	QA pattern	Qtarget	Window	Total
78.1	05.5	26.2	10.4	30.3

Table 4. TREC-9 test: correct answers attributable to each module.

We are pleased with the performance of the Qtarget match. This shows that CONTEX is able to identify to some degree the semantic type of the desired answer, and able to pinpoint these types also in candidate answers. The fact that it outperforms the window match indicates the desirability of looking deeper than the surface level. As discussed in Section 4, we are strengthening the parser's ability to identify Qtargets.

We are disappointed in the performance of the 500 QA patterns. Analysis suggests that we had too few patterns, and they were too specific. When patterns matched, they were very accurate, both in finding correct answers and pinpointing the exact boundaries of answers. But they were too sensitive to variations in phrasing. Furthermore, it was difficult to construct robust and accurate question and answer phraseology patterns manually, for several reasons. First, manual construction relies on the inventiveness of the pattern builder to foresee variations of phrasing, for both question and answer. It is however nearly impossible to think of all possible variations when one builds a pattern. Second, it is not always clear at what level of representation to formulate the pattern: when should one specify using words? Parts of speech? Other parse tree nodes? Semantic classes? The patterns in Figure 3 include only a few of these alternatives. Specifying the wrong elements can result in non-optimal coverage. Third, the work is simply tedious. We therefore decided to try to learn QA patterns automatically.

7. Toward Learning QA Patterns Automatically

To learn corresponding question and answer expressions, we pair up the parse trees of a question and (each one of) its answer(s). We then apply a set of matching criteria to identify potential corresponding portions of the trees. We then use the EM algorithm to learn the strengths of correspondence combinations at various levels of representation. This work is still in progress.

In order to learn this information we observe the truism that there are many more questions than answers. This holds for the two QA corpora we have access to—TREC and an FAQ website (since discontinued). We therefore use the familiar version of the Noisy Channel Model and Bayes' Rule. For each basic QA type (Location, Why-Famous, etc.):

$$P(A|Q) = \text{argmax } P(Q|A) \cdot P(A)$$

$$P(A) = \sum_{\text{all trees}} (\text{number of nodes that may express a true A}) / (\text{number of nodes in tree})$$

$$P(Q|A) = \sum_{\text{all QA tree pairs}} (\text{number of covarying nodes in Q and A trees}) / (\text{number of nodes in A tree})$$

As usual, many variations are possible, including how to determine likelihood of expressing a true answer; whether to consider all nodes or just certain major syntactic ones (N, NP, VP, etc.); which information within each node to consider (syntactic? semantic? lexical?); how to define 'covarying information'—node identity? individual slot value equality?; what to do about the actual answer node in the A trees; if (and how) to represent the relationships among A nodes that have been found to be important; etc. Figure 3 provides an answer parse tree that indicates likely Location nodes, determined by appropriate syntactic class, semantic type, and syntactic role in the sentence.

Our initial model focuses on bags of corresponding QA parse tree nodes, and will help to indicate for a given question what type of node(s) will contain the answer. We plan to extend this model to capture structured configurations of nodes that, when matched to a question, will help indicate where in the parse tree of a potential answer sentence the answer actually lies. Such bags or structures of nodes correspond, at the surface level, to important phrases or words. However, by using CONTEX output we abstract away from the surface level, and learn to include whatever syntactic and/or semantic information is best suited for predicting likely answers.

8. References

Bikel, D., R. Schwartz, and R. Weischedel. 1999. An Algorithm that Learns What's in a Name. *Machine Learning—Special Issue on NL Learning*, 34, 1–3.

Choi, F.Y.Y. 2000. Advances in independent linear text segmentation. In *Proceedings of the 1st Meeting of the North American Chapter of the Association for Computational Linguistics* (ANLP-NAACL-00), 26–33.

Fellbaum, Ch. (ed). 1998. *WordNet: An Electronic Lexical Database.* Cambridge: MIT Press.

Gerber, L. 2001. A QA Typology for Webclopedia. In prep.

Hearst, M.A. 1994. Multi-Paragraph Segmentation of Expository Text. *Proceedings of the Annual Meeting of the Association for Computational Linguistics* (ACL-94).

Hermjakob, U. 1997. *Learning Parse and Translation Decisions from Examples with Rich Context*. Ph.D. dissertation, University of Texas at Austin. file://ftp.cs.utexas.edu/pub/ mooney/papers/hermjakob-dissertation-97.ps.gz.

Hermjakob, U. 2000. Rapid Parser Development: A Machine Learning Approach for Korean. In *Proceedings of the North American chapter of the Association for Computational Linguistics* (NAACL-2000). http://www.isi.edu/~ulf/papers/kor_naacl00.ps.gz.

Hermjakob, U. 2001. Parsing and Question Classification for Question Answering. In prep.

Hovy, E.H., L. Gerber, U. Hermjakob, M. Junk, and C.-Y. Lin. 2000. Question Answering in Webclopedia. *Proceedings of the TREC-9 Conference*. NIST. Gaithersburg, MD.

Moldovan, D., S. Harabagiu, M. Pasca, R. Mihalcea,, R. Girju, R. Goodrum, and V. Rus. 2000. The Structure and Performance of an Open-Domain Question Answering System. In *Proceedings of the Conference of the Association for Computational Linguistics* (ACL-2000), 563–570.

Srihari, R. and W. Li. 2000. A Question Answering System Supported by Information Extraction. In *Proceedings of the 1st Meeting of the North American Chapter of the Association for Computational Linguistics* (ANLP-NAACL-00), 166–172.

Witten, I.H., A. Moffat, and T.C. Bell. 1994. Managing Gigabytes: Compressing and indexing documents and images. New York: Van Nostrand Reinhold.

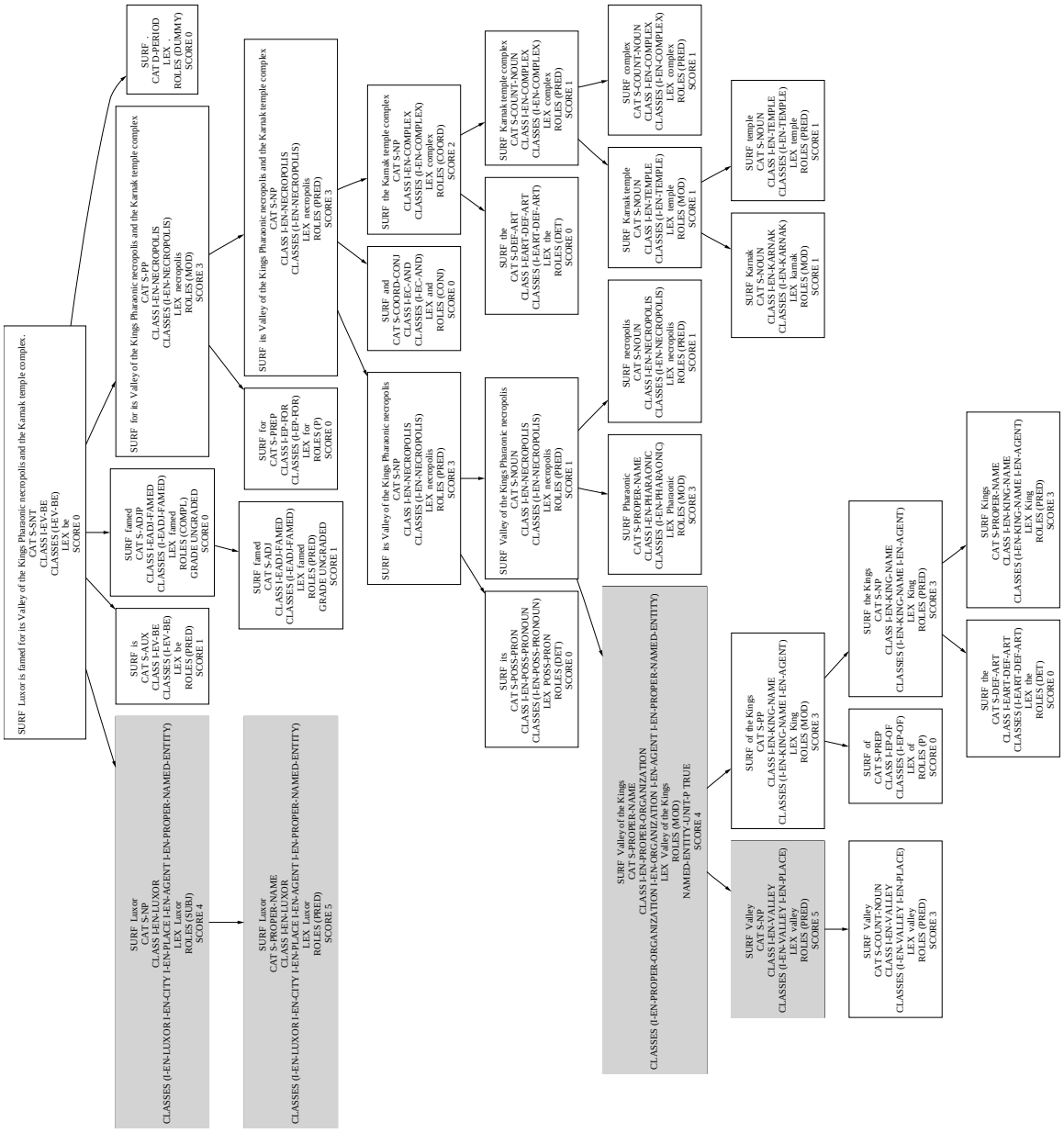

Figure 3. Candidate answer tree showing likely Location answers.