INTERSPEECH 2009 BRIGHTON

Combining Semantic and Syntactic Information Sources for 5-W Question Answering

Sibel Yaman[1], *Dilek Hakkani-Tur*[1], *Gokhan Tur*[2],

[1] International Computer Science Institute,
[2] SRI International
{sibel,dilek}@icsi.berkeley.edu,
gokhan@speech.sri.com

Abstract

This paper focuses on combining answers generated by a semantic parser that produces semantic role labels (SRLs) and those generated by syntactic parser that produces function tags for answering 5-W questions, i.e., *who, what, when, where,* and *why*. We take a probabilistic approach in which a system's ability to correctly answer 5-W questions is measured with the likelihood that its answers are produced for the given word sequence. This is achieved by training statistical language models (LMs) that are used to predict whether the answers returned by semantic parse or those returned by the syntactic parser are more likely. We evaluated our approach using the OntoNotes dataset. Our experimental results indicate that the proposed LM-based combination strategy was able to improve the performance of the best individual system in terms of both F_1 measure and accuracy. Furthermore, the error rates for each question type were also significantly reduced with the help of the proposed approach.

Index Terms: Question answering, Spoken language understanding applications

1. Introduction

The goal in the third year of DARPA-funded GALE Information Distillation task is to extract *exact* answers to 5-W questions, i.e., *who, what, when, where,* and *why*, for a top-level predicate of each given sentence. The motivation for the 5-W task is that the answers to 5-W questions cover the basic information nuggets in a sentence. If a system can isolate these pieces of information successfully, then it can produce the basic meaning of the sentence.

The GALE Information Distillation task specifies what kinds of constituents are expected in each answer. According to the specifications, the logical subject of the chosen predicate should answer *who*. It will be *null* whenever there is no subject, for instance, in imperative sentences or in passive sentences in which there is no logical subject indicated with a "by clause". The predicate itself and its logical object should appear in the answer to *what*. The temporal argument of the main predicate is expected to answer *when*. It may refer to specific times such as "on Monday", non-exact times such as "after the war", and adverbs of frequency and duration such as "always". Physical locations such as "in the building" as well as metaphorical locations such as "in his speech" are expected as answers to *where*. Explicitly triggered expressions of reason, cause, and purpose such as "as a result of the crisis" and "because of the explosion" should answer *why*. The returned answers are judged "*correct*" only if they either correctly identify a null answer (i.e., there is no answer to be returned) or correctly extract an answer that is present in the sentence. Answers are not penalized for including extra text provided that the extra text is not from another answer or from another top-level predicate.

It has been observed that incorrect parses account for a major part of the errors in similar question answering systems. This is mainly because the ability of a system to answer given questions is limited by the depth of the available semantic representations. Another reason is that most systems make the strong assumption that the input will be a typical sentence. A better approach would therefore take atypical cases (such as speech recognition errors) into account, however, due to mostly unpredictable nature of atypical cases this approach is of limited use.

In this paper, we have two 5-W answer generating modules: one that relies on syntactic parses with function tags (such as "subject", "location", and so on), and another that relies on semantic parses with semantic role labels (SRLs) that indicate predicate-argument relations (ARG0 through ARG5 and ARGMs) [1, 2]. Our combination strategy receives answers from these two modules and codes them as answer sequences (e.g., "$<s>$ ω_1*_who* ω_2*_who* ω_3*_what* ω_4*_what* ω_5*_what* ω_6*_when* $</s>$") for the word sequence ($\omega_1, ..., \omega_6$. We then combine the answers generated by these two modules using a language model (LM)-based strategy. An LM is trained to estimate the probabilities of n-gram answer sequences (e.g., $Pr(who|who)$, $Pr(what|who)$, etc.). Another LM is trained on manually generated answers to predict the probability of an answer given the word sequence (e.g., $Pr(< word > |who)$, $Pr(< word > |what)$. A score that combines these two LMs is computed for the answers of each of the modules and the 5-W answers of the higher scoring module are accepted. As an alternative approach to the problem of multiple 5W answers is presented in [3]. The authors develop three independent systems and extract useful features from the answers returned by each answer. Then the answers are rescored based on SVM classification scores. An approach that is similar to ours was taken in [4] for speech understanding. It was based on the stochastic modeling of a sentence as a sequence of elemental units that represent its meaning.

In this paper, we use hand-corrected parse trees of text genres to train models, whereas use automatic parse trees of closed-caption speech transcriptions as test data. Note that closed-caption speech transcriptions also consists of many errors such as the shortening of long sentences, skipping details, and so on. These errors may result in significant deterioration in the quality of automatic parses. One other factor that deteriorates the quality of automatic parses is that disfluencies (e.g., "*um*", repeated words, repairs, false starts, phrases like "*you know*" and so on)

are common in normal speech. We designed our systems so that disfluencies are removed with the help of syntactic information as much as possible. Our experimental results indicate a significant improvement over the best performing system in terms of reduced error rate and increased F_1 measure.

2. 5-W Question Answering Task

The goal in the third year of DARPA-funded GALE Information Distillation task is to extract *exact* answers to the 5-W questions, i.e., *who, what, when, where,* and *why*, for a top-level predicate of each given sentence. Multiple plausible top-level predicates exist in many cases including conjunctions or multiple independent clauses (e.g., "*Bob went and bought drinks*"), and statements (e.g., "*Bob went to the store, police said*"). Selecting either predicate is considered "correct" in such cases.

Each non-null answer, a, returned by a system is regarded correct if both of the following statements are true: ((i) a is of the appropriate type (e.g. *when*, *where*, etc.), (ii) a could plausibly be an argument of a top-level predicate in the sentence. Each null answer returned by the system is "correct" if there is no argument of this type that clearly modifies any of the top-level predicate in the reference.

Answers are not penalized for including extra text, such as prepositional phrases or subordinate clauses. However, if the extra text includes text from another answer or text from another top-level predicate, the answer is considered "incorrect". Answers may also be judged "partial", meaning that only part of the answer was returned. For example, if the the answer to *what* contains the predicate but not the logical object, it is graded as "partial".

As an illustration, consider the sentence "*There were elections held, I believe, a year or so ago*" for which there are more than one set of acceptable answers as follows:

- **Answer 1:** *Who*: I, *What*: believe there were elections held a year or so ago.
- **Answer 2:** *Who*: There, *What*: were elections held a year or so ago.
- **Answer 3:** *Who*: There, *What*: were elections held, *When*: a year or so ago.
- **Answer 4:** *What*: were elections held a year or so ago, *Where*: there.
- **Answer 5:** *What*: were elections held, *When*: a year or so ago, *Where*: there.

2.1. Using Syntactic Parses with Function Tags

Our first 5-W question answering system uses the syntactic parses with function tags produced by the University of Maryland parser that was specifically designed to handle ASR outputs (see [5, 3] for details) and works in two steps. The first step is a cascade of several operations to determine one top-level predicate: detecting and marking quotations, removing conditional clauses, processing conjunction of sentences and conjunction of verb phrases so that only one top-level predicate remains, and detecting passive sentences. The second step starts with an analysis of the sentence structure since the positions of constituents in the sentence depend on the sentence structure. For instance, the subject is before the verb phrase in simple active declarative sentences but it is after the verb phrase in inverted sentences.

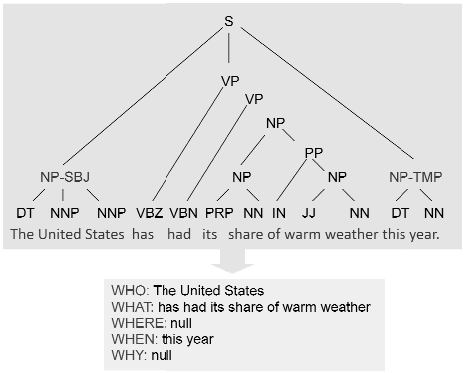

Figure 1: The syntactic parse trees with function tags (e.g. "SBJ", "TMP", "LOC", "DIR", and "PRP") are used to identify exact answers to 5-W questions.

Once the sentence is categorized according to its structure, a set of linguistically-motivated handcrafted rules is applied to extract 5-W answers. These rules assign the constituents (that are attached to the top-level predicate) with tags "SBJ" and "LGS" to *who*, those with tag "'TMP" to *when*, those with tags "LOC" and "DIR" to *where*, and those with tag "PRP"' to *why*. The answer to *what* is found by further processing of the remaining constituents so that optional phrases are filtered out and no repetitions of answers occur. An example of the mapping from syntactic parses with function tags into 5-W answers is shown in Figure 1 for the sentence "*The United States has had its share of warm weather this year*".

2.2. Using Semantic Role Labels

When presented with a sentence, a semantic role labeler identifies and labels the semantic arguments of each of the predicates in the sentence. In the ASSERT semantic role labeler [6], which we used in this work, this is achieved by extracting features for all constituents in the parse tree relative to the predicate. The constituents are then classified into one class of semantic role label using one-versus-all binary support vector machines (SVMs) using the manually annotated PropBank data.

The permissible semantic arguments depend not only the predicate itself but also the sense a predicate is being used in the sentence. Generally speaking, ARG0 stands for agent, ARG1 for the theme or direct object, and ARG2 for indirect object, benefactive or instrument. Additionally, predicates might have adjunctive arguments, referred to as ARGMs. For example, ARGM-LOC indicates a locative, ARGM-TMP indicates a temporal, and ARGM-CAU indicates a causative argument. As in our first system, our second 5-W question answering system first process the sentence so that only one top-level predicate is considered and its category (declarative, imperative, and so on) is determined. Afterwards, the semantic role labels are mapped into answers. Note that this requires predicate-specific considerations as there are some exceptional cases; for instance, ARG3 represents a locative argument (and hence answers where) for the predicate "go". An example for mapping SRLs into 5-W answers is shown in Table 1 for the above sentence.

3. Combining Multiple 5-W Answer Extractors Using Language Models

In the related problem of combining multiple semantic role labeling modules, it has been found that an argument combina-

Table 1: Mapping SRLs into 5-W answers

ARG0	The United States
Target+ARG1	has had its share of warm weather
ARGM-TMP	this year

tion strategy that may return different arguments from different modules is better than a combination strategy that takes arguments from one of the modules [7]. In the task that we consider, the individual modules are required to return arguments for only one of the predicates and they are free to select which predicate to consider. Since the arguments are dependent on the predicate chosen, it is not wise in our problem to return different arguments from different modules. Furthermore, as our experimental results show, the systems we developed make complementary mistakes and therefore the ideal combination strategy that selects the best system for each sentence was able to give an accuracy of as much as 95% overall.

The 5-W answer sequence for the permissible answer "Answer 2" of the example sentence of Section 2 would be "<s> who what what what else else what what what what what < /s>". This 5-W answer sequence corresponds to marking "there" with *who*, "were" with *what*, and so on. Therefore, the task of finding the 5-W answers can be modeled as that of predicting the probability that the given word sequence (i.e., sentence) produces the given 5-W answer sequence.

An LM is useful to find answers to 5-W questions in several ways. An LM that models the probability of consecutive 5-W answers restricts the permissible 5-W answer sequences that are specific to language and genre. For instance, the sequence "<s> who ... who what...what < /s>" is the predominant pattern in simple (grammatical) declarative sentences such as "The president will give a talk". However, a sequence like "<s> who ... who what ... what .... who ... who < /s>" is not a permissible answer sequence for simple (grammatical) declarative sentences.

An LM that models the probability of words being produced by a given 5-W answer estimates how likely that a word will appear in a given 5-W answer. It estimates, for instance, that the word sequence "there" will likely to appear in the answer to *where* (as in "It is raining there") or *who* (as in "There is rain"). Due to the probabilistic nature of LMs, some probability is still reserved for the case that, for instance, "there" will appear in the answer to "when" (consider, "The president will give a talk when he arrives there").

Let $W = \{\omega_1, \omega_2, ..., \omega_N\}$ denote a sequence of words and $Q = \{q_1, q_2, ..., q_N\}$ denote a sequence of 5-W answer sequence, where q denotes a question type from the set $\{"who", "what", "when", "where", "why", "else"\}$. The most likely 5-W answer sequence is the one that gives the highest probability of $P(Q|W)$, i.e.,

$$\hat{Q} = \arg\max_{Q} Pr(Q|W) \quad (1)$$

$$= \arg\max_{Q} Pr(W|Q)P(Q) \quad (2)$$

The first term of Eqn. (1), $P(W|Q)$, can be approximated as

$$P(W|Q) = \prod_i Pr(\omega_i|q_i, \omega_{i-1}, \omega_{i-2}) \quad (3)$$

using a trigram model for word sequences and an independence assumption that the i^{th} word is independent of the question types the previous words answer (i.e., q_1 through q_{i-1}) given the question type it answers (i.e., q_i). The probability $Pr(\omega_i|q_i, \omega_{i-1}, \omega_{i-2})$ estimates the likelihood that ω_i will appear in the answer to question q_i given its contextual words, ω_{i-1} and ω_{i-2}.

The second term of Eqn. (1), $P(Q)$, is the prior probability of a given 5-W answer sequence and can be approximated using a trigram language model as

$$P(Q) = \prod_i P(q_i|q_{i-1}, q_{i-2}) \quad (4)$$

Hence, a decision is made in favor of the system which yields a greater value of $Pr(Q|W)$.

4. Experiments and Results

We ran experiments to evaluate the proposed approach and compare their performance against individual systems and against an oracle system. In this section, we first describe the corpus that we use and then report our experimental findings.

4.1. Semantic Corpus

The OntoNotes corpus comprises of various genres of text that are annotated with structural information (syntax and predicate argument structure) and shallow semantics (word sense linked to an ontology and coreference) [8]. The OntoNotes corpus provides a wealth of resources; however, finding answers to the 5-W questions for one top-level predicate introduces many challenges for preparing a corpus that would be useful. Since each sentence may have multiple correct sets of 5-Ws, it is not straightforward to produce a gold-standard corpus for automatic evaluation.

To combat such challengesin producing a gold standard, we took a two-step approach to data preparation. In the first step, we automatically produced 5-W answers using the rules that find answers from syntactic parses with function tags. These were then hand-corrected so that they can be used for training language models. In the meantime, the answers of the individual modules were compared against the hand-corrected answers and judged as "correct", "incorrect", or "partial". These judgments were also hand-corrected to account for the fact that there may be more than one set of correct answers. The manual annotations of OntoNotes are used to train LMs. Our test data consisted of automatic syntactic parses generated by the UMD parser using closed-captions of the speech data. Eventually, our training data consisted of 566 sentences, and our test data consisted of 406 sentences.

4.2. Overall Performance Evaluation

Table 2 shows the error rates of different systems and combination strategies on different genres. "F-Tag" stands for the system in which answers are found from syntactic parses with function tags, and "SRL" stands for the system in which answers are found from semantic role labels. The "Oracle" stands for the ideal strategy that has the knowledge of which system would perform best for a given sentence, and hence denotes the best that can be expected from any combination scheme. The results in the "LM" row show the performance obtained using the LM-based approach of Section 3.

As seen From Table 2, in the overall, SRL is 30% worse than F-Tag answer extraction. In the meantime, the error rate of the oracle system is as low as 4.5%, which suggests that the perfect combination strategy would cut the error rate by half. The proposed LM-based combination strategy is able to reduce

Table 2: The error rates per question type in 5-W question answering.

Accuracy	*who*	*what*	*when*	*where*	*why*	all
F-Tag	8.4	19.5	5.7	9.4	3.0	9.2
SRL	10.6	27.1	8.2	10.6	3.7	12.0
LM	6.7	17.0	6.0	8.9	2.3	8.2
Oracle	4.7	10.1	2.8	3.5	1.5	4.5

Table 3: The precision, recall, and F_1 measures

	Precision	Recall	F_1
F-Tag	91.2	81.5	82.6
SRL	91.1	79.9	82.5
LM	89.9	85.8	85.5
Oracle	96.6	92.4	92.9

the error rate to 8.2%, which means a relative error reduction of 11% compared F-Tag.

In Table 3, the performance is given in terms of precision (P), recall (R), and F_1 measure, which turned out to be comparable. The LM-based combination strategy was able to increase the F_1 measure by roughly 3 figures, which is statistically significant.

It is important to note that evaluation in terms of error rates is a better criterion in this task than evaluation in terms of P,R, or F_1. Consider the example sentence of Section 2, i.e., "*There were elections held, I believe, a year or so ago*". As mentioned there, there are more than one set of correct answers for this sentence. For instance, "a year or so ago" could be thought of as attached to "held" or to "were". In the former case, it would appear in the answer to *what*, whereas in the second case, it would appear in the answer to *when*. Therefore, both "null" and "a year or so ago" would be acceptable answers to *when*. When calculating P, R, or F_1, this situation would represent an inconsistency. However, the computation of error rates only consider the given "correct" and "incorrect" annotations without any distinction as to type-I and type-II errors.

4.3. Performance Evaluation per Question Type

It is apparent from Figure 2 that the SRL system performs worse than the F-Tag system in all question types. The difference is significant in all question types. In *when* SRL performs about relatively 43% worse than F-Tag, and in *what*, about relatively 39% worse. The most important reason was that the SRL component quite often fails to return argument for any of the predicates in the answers, which meant all-null answers, whereas the F-Tag component always produced an output. A back-up procedure would allow the SRL component to avoid many of the all-null answers. Another reason was that the SRL component returned less non-null answers for *when* and *where*, which seem to have worked to its disadvantage.

As Table 2 suggests, the performance on the answers to *what* is substantially lower than those answers to other questions. Just to highlight the performance differences across question types, Figure 2 shows the results in Table 2 visually. One of the most important reasons is that the answers to *what* tend to contain extra text, which quite often contains arguments of a predicate other than the top-level predicate. It was also quite common that the answer to *what* contained text that would correctly answer other questions. For instance, if the answer to *when* is contained in the answer to what, then the answers to both *when* and *what* are "incorrect".

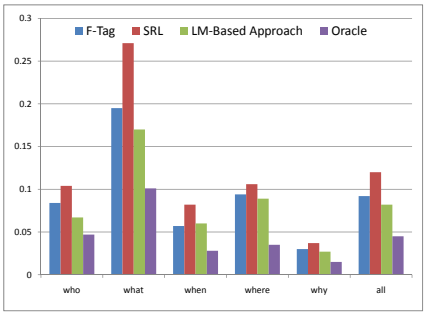

Figure 2: The error rates per question type for each system (from left to right: F-Tag, SRL, LM-Based Combination Approach, and Oracle.

5. Conclusions

In this paper, we described two systems to generate answers to 5-W questions: one relies on a semantic parser that produces semantic role labels (SRLs) and the other that relies on a syntactic parser that produces function tags. We propose a language model-based approach to combining these two systems and analyze why such a combination strategy is taken. Specifically, a system's ability to correctly answer 5-W questions is measured with the probability of its answers being produced for the given word sequence. We propose training statistical language models (LMs) to predict whether the answers returned by semantic or by syntactic parsers are more likely. We evaluated our approach using the OntoNotes dataset. Our experimental results indicate that the proposed LM-based combination strategy was able to improve the performance of the best individual system in terms of both the F_1 measure and the error rate. Furthermore, the error rates for each question type were also significantly reduced.

6. Acknowledgements

This work was supported by DARPA HR0011-06-C-0023. Any opinions, findings and/or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.

7. References

[1] D. Gildea and D. Jurafsky, "Automatic labeling of semantic roles," *Computational Linguistics*, vol. 28, no. 3, pp. 245–288, 2002.

[2] S. Narayanan and S. Harabagiu, "Question answering based on semantic structures," in *COLING*, 2004.

[3] S. Yaman, D. Hakkani-Tur, G. Tur, R. Grishman, M. Harper, K. R. McKeown, A. Meyers, and K. Sharma, "Classification-based strategies for combining multiple 5-w question answering systems," in *InterSpeech*, 2009.

[4] E. Levin and R. Pieraccini, "Concept-based spontaneous speech understanding system," in *EuroSpeech*, 1995.

[5] M. Harper, B. Dorr, J. Hale, B. Roark, I. Shafran, M. Lease, Y. Liu, M. Snover, L. Yung, A. Krasnyanskaya, and R. Stewart, "Parsing and spoken structural event," The Johns-Hopkins University, 2005 Summer Research Workshop, Tech. Rep., 2005.

[6] S. S. Pradhan, W. Ward, K. Hacioglu, J. H. Martin, and D. Jurafsky, "Shallow semantic parsing using support vector machines," in *HLT/NAACL*, 2004.

[7] M. Surdeanu, L. Marquez, X. Carreras, and P. R. Comas, "Combination strategies for semantic role labeling," *Journal of Artificial Intelligence Research*, vol. 29, pp. 105–151, 2007.

[8] E. Hovy, M. Marcus, M. Palmer, L. Ramshaw, and R. Weischedel, "OntoNotes: The 90% solution," in *HLT/NAACL*, 2006.