DOI:10.1145/2483852.2483869

The RSVP voice-recognition search engine improves speech recognition and translation accuracy in question answering.

BY YANG TANG, DI WANG, JING BAI, XIAOYAN ZHU, AND MING LI

Information Distance Between What I Said and What It Heard

VOICE INPUT IS a major requirement for practical question answering (QA) systems designed for smartphones. Speech-recognition technologies are not fully practical, however, due to fundamental problems (such as a noisy environment, speaker diversity, and errors in speech). Here, we define the information distance between a speech-recognition result and a meaningful query from which we can reconstruct the intended query, implementing this framework in our RSVP system.

In 12 test cases covering male, female, child, adult, native, and non-native English speakers, each with 57 to 300 questions from an independent test set of 300 questions, RSVP on average reduced the number of errors by 16% for native speakers and by 30% for non-native speakers over the best-known speech-recognition software. The idea was then extended to translation in the QA domain.

In our project, which is supported by Canada's International Development Research Centre (http://www.idrc.ca/), we built a voice-enabled cross-language QA search engine for cellphone users in the developing world. Using voice input, a QA system would be a convenient tool for people who do not write, for people with impaired vision, and for children who might wish their Talking Tom or R2-D2 really could talk.

The quality of today's speech-recognition technologies, exemplified by systems from Google, Microsoft, and Nuance does not fully meet such needs for several reasons:

- Noisy environments in common audio situations;[1]
- Speech variations, as in, say, adults vs. children, native speakers vs. non-native speakers, and female vs. male, especially when individual voice-input training is not possible, as in our case; and
- Incorrect and incomplete sentences; even customized speech-recognition systems would fail due to coughing, breaks, corrections, and the inability to distinguish between, say, "sailfish" and "sale fish."

Speech-recognition systems can be trained for a "fixed command set" of up to 10,000 items, a paradigm that

» key insights

- **Focusing on an infinite but highly structured domain (such as QA), we significantly improve general-purpose speech recognition results and general-purpose translation results.**
- **Assembling a large amount of Internet data is key to helping us achieve these goals; in the highly structured QA domain, we collected millions of human-asked questions covering 99% of question types.**
- **RSVP development is guided by a theory involving information distance.**

does not work for general speech recognition. We consider a new paradigm: speech recognition limited to the QA domain covering an unlimited number of questions; hence it cannot be trained as a fixed command set domain. However, our QA domain is highly structured and reflects clear patterns.

We use information on the Internet in the QA domain to find all possible question patterns, then use it to correct the queries that are partially recognized by speech-recognition software. We collected more than 35 million questions from the Internet, aiming to use them to infer templates or patterns to reconstruct the original intended question from the speech-recognition input. Despite this, we must still address several questions:

- How do we know if an input question from the speech-recognition system is indeed the original user's question?;
- How do we know if a question in our database is the user's intended question?;
- Should we trust the input or the database?; and
- Often, neither the database nor the input is always exactly right, so can we reconstruct the original question?

We provide a mathematical framework to address these questions and implement the related RSVP system. Our experiments show RSVP significantly improves current speech-recognition systems in the QA domain.

Related Work

Speech recognition[1] has made significant progress over the past 30 years since the introduction of statistical methods and hidden Markov models. Many effective algorithms have been developed, including the EM algorithm, the Baum-Welch algorithm, Viterbi N-best search, and N-gram language models trained on large data corpora. However, as explored by Baker et al.,[1] automatic speech recognition is still an unsolved problem.

Unlike traditional speech-recognition research, we propose a different paradigm in the QA domain. In it, we are able to collect a very large pure text corpus (no voice) that differs from the fixed command set domain where it is possible to train up to, say, 10,000 commands. The QA domain is unbounded, and the number of existing questions on QA websites involves more than 100 million questions, yet with very low coverage of all possible questions. These texts can be "clustered" into patterns. Here, we demonstrate that these patterns have 99% coverage of all possible question types, suggesting we can use them to improve speech-recognition software in this domain.

Previous research suggested that context information,[21] the knowledge-base,[16,20] and conceptual relationships[15] all can help address this.

Information Distance

Here, we develop the mathematical framework on which we designed our system. To define Kolmogorov complexity (invented in the 1960s), we start by fixing a universal Turing machine U. The Kolmogorov complexity of a binary string x, given another binary string y, $K_U(x|y)$, is the length of the shortest (prefix-free) program for U that outputs x with input y. Since it can be shown that for a different universal Turing machine U', the metric differs by only a constant, we write $K(x|y)$ instead of $K_U(x|y)$. We write $K(x|\varepsilon)$, where ε is the empty string, as $K(x)$. We call a string x random if $K(x) \geq |x|$. See Li and Vitányi[14] for more on Kolmogorov complexity and its rich applications.

Note $K(x)$ defines the amount of information in x. What would be a good departure point for defining an "information distance" between two objects? In the 1990s, Bennett et al.[2] studied the energy cost of conversion between two strings, x and y. John von Neumann hypothesized that performing 1 bit of information processing costs $1KT$ of energy, where K is Boltzmann's constant and T is the room temperature. In the 1960s, observing that reversible computations could be done at no cost, Rolf Landauer revised von Neumann's proposal to hold only for irreversible computations. Starting from this von Neumann-Landauer principle, Bennett et al.[2] proposed using the minimum number of bits needed to convert x to y and vice versa to define their distance. Formally, with respect to a universal Turing machine U, the cost of conversion between x and y is defined as

$$E(x,y) = \min\{|p| : U(x,p) = y, U(y,p) = x\} \quad (1)$$

It is clear that $E(x,y) \leq K(x|y) + K(y|x)$. Bennett et al.[2] obtained the following optimal result, modulo $\log(|x| + |y|)$:

Theorem 1. $E(x,y) = \max\{K(x|y), K(y|x)\}$.

Thus, we define information distance between two sequences, x and y, as $D_{\max}(x,y) = \max\{K(x|y), K(y|x)\}$.

This distance $D_{\max}$ was shown to satisfy the basic distance requirements (such as positivity, symmetricity, and triangle inequality). Furthermore, $D_{\max}$ is "universal" in the sense that it always minorizes any other reasonable computable distance metric.

We use information on the Internet in the QA domain to find all possible question patterns, then use it to correct the queries that are partially recognized by speech-recognition software.

This concept, and its normalized versions, were applied to whole-genome phylogeny,[12] chain-letter-evolution history,[3] plagiarism detection,[4] other phylogeny studies,[13] music classification,[5] and parameter-free data mining,[10] and has been followed by many other applications (for topics mentioned here that do not have references, see Li and Vitányi[14]) including protein-structure comparison, heart-rhythm data analysis, QA systems, clustering, multiword expression linguistic analysis, software evolution and engineering, software metrics and obfuscation, webpage authorship, topic, and domain identification, phylogenetic reconstruction, SVM kernel for string classification, ortholog detection,[19] analyzing worms and network traffic, image similarity, Internet knowledge discovery,[6] multidocument summarization, network structure, and dynamic behavior,[17] and gene expression dynamics in macrophase.[18]

Despite its useful properties and applications, the max distance $D_{\max}(x,y)$ involves several problems when only partial matches are considered[8,22] where the triangle inequality fails to hold and irrelevant information must be removed. Thus, Li et al.[11] introduced a complementary information-distance metric to resolve these problems. In Equation 1 we determine the smallest number of bits that must be used to reversibly convert between x and y. To remove the irrelevant information from x or y, we thus define, with respect to a universal Turing machine U, the cost of conversion between x and y as

$$E_{\min}(x,y) = \min\{|p| : U(x,p,r) = y, U(y,p,q) = x, |p| + |q| + |r| \leq E(x,y)\}, \quad (2)$$

This definition separates r from x and q from y. Modulo an $O(\log(|x| + |y|))$ additive term, the following theorem was proved in Li[11]:

Theorem 2. $D_{\min}(x,y) = \min\{K(x|y), K(y|x)\}$.

We can thus define $D_{\min}(x,y) = E_{\min}(x,y)$ as a complementary information-distance metric that disregards irrelevant information. $D_{\min}$ is obviously sym-

metric but does not satisfy triangle inequality. Note that $D_{\min}$ was used in the QUANTA QA system to deal with concepts that are more popular than others.[23,24]

Min-Max Distance

Now we formulate our problem in the frame of information distance, given a question database Q and k input questions from a speech-recognition system, as in $I = \{q_1,....,q_k\}$ ($k \leq 3$ for the Google speech-recognition server in our experiments and $k = 1$ in our translation application. The goal is to compute the user's intended question q. It could be one of the q_is; it could be a combination of all k of them; and it could also be one of the questions in Q that is close to some parts of the q_is.

We wish to find the most plausible question q, such that q fits one of the question patterns in Q, and q is "close" to I. We assume Q contains almost all question patterns; later, we provide an experimental justification for this claim.

We can thus formulate our problem as: Given Q and I, find q such that it minimizes the sum of "distances" from Q to q and q to I, as in

$$I \longleftrightarrow q \longleftrightarrow Q$$

Here, Q is a huge database of 35 million questions asked by users. We assume q is "similar" to one of them; for example, a QA user might ask "Who is the mayor of Waterloo, Ontario?," but Q might include such questions as "Who is the mayor of Toronto, Ontario?" and "Who is the mayor of Washington, D.C.?" I sometimes contains questions like "Hole is the mayor?" and "Who mayor off Waterloo" from the speech-recognition software. Since Q is so large, the $D_{\max}$ measure does not make sense here, as most of the information in Q is irrelevant. It is natural to use $D_{\min}(q,Q)$ here. For the distance between q and I, we use $d_{\max}(q, I)$ to measure it. Given I, Q, we wish to find q that minimizes the function

$$\delta D_{\min}(q,Q) + D_{\max}(I,q), \qquad (3)$$

where $D_{\min}$ measures information distance between q and Q with irrelevant information removed; $D_{\max}$ is the information distance between q and I. We know

$$D_{\min}(x,y) = \min\{K(x|y), K(y|x)\},$$
$$D_{\max}(x,y) = \max\{K(x|y), K(y|x)\}.$$

Thus, $D_{\min}(q,Q) = K(q|Q)$, because Q is very large and q is a single question. Note that d is a coefficient that determines how much weight we wish to give to a correct template or pattern in Q.

Equation 3 thus becomes

$$\delta K(q|Q) + \max\{K(q|I), K(I,q)\}. \qquad (4)$$

Obervations: We need $\delta > 1$, so $q = I$ does not minimize Equation 4. If δ is too large, then $q = \varepsilon$ might minimize Equation 4. There is a trade-off: Sometimes a less-popular pattern (taking more bits in the $D_{\min}$ term) might fit I better (taking fewer bits in the $D_{\max}$ item), and a more popular pattern (taking fewer bits in the $D_{\min}$ item) might miss one or two key words in I, taking more bits to encode in the $D_{\max}$ item. Note that δ is optimized for the trade-off.

Encoding Issues

To minimize Equation 4, we solve three problems:

- Encode q using Q in the first term. It is a problem to encode an item with respect to a big set;
- Encode q using I or encode I using q, and take whichever is larger in the second term; and
- Find all possible candidates q and a q_0 that minimizes Equation 4.

We see that Q is very large and contains different "types" of questions. For each such type, we extract one or more question templates. In this way, Q can be viewed as a set of templates, with each template, denoted as p, covering a subset of questions from Q. When encoding q, we need not encode q from Q directly. Instead, we encode q with respect to the patterns or templates of Q; for example, if a pattern p in Q appears N times in Q, then we use $\log_2(|Q|=N)$ bits to encode the index for this pattern. Given pattern p, we encode q with p by encoding their word mismatches. There is a trade-off between the encoding of p and the encoding of q, given p. A common pattern may be encoded with a few bits but also may require more bits to encode a specific question using the pattern; for example, the template "who is the mayor of City Name" requires more bits to encode than the template "who is the mayor of Noun" because the former is a smaller class than the latter. However, the first template requires fewer bits to generate the question "who is the mayor of Waterloo" since it requires fewer bits to encode Waterloo from the class "City Name" than from the class "Noun."

The patterns could be extracted by pre-processing or dynamically according to the input. In practice, we extract patterns only from questions relevant to I, denoted as Q'. We organize Q' hierarchically. Similar questions are mapped to a cluster, and similar clusters are mapped to a bigger cluster. We extract one pattern from each cluster using a multiple alignment algorithm. This pattern should be as specific as possible while at the same time cover all questions in the cluster. Note that the higher the cluster in the hierarchical structure, the more general the pattern. Our hierarchical clustering algorithm thus assures we can extract all possible patterns from relevant questions. We make use of numerous semantic and syntactic information sources during the process, including POS tagger, Name Entity Recognition, WordNet, and Wikipedia. For example, given a cluster of three questions:

- Who is the mayor of Toronto?;
- Who is the president of the United States?; and
- Who is a senator from New York?

We could extract one pattern (such as Who is the Leader of Location? "Mayor," "president," and "senator") are all mapped to the Leader class, while "Toronto," "United States," and "New York" all belong to the Location class.

If we treat pattern p as a sentence, the problem of item-to-set encoding becomes item-to-item encoding, as in the computation of $K(q|I)$ and $K(I|q)$. To convert a sentence from another sentence, we need encode only the word mismatches and the missing words. The best alignment between two sentences is found through a standard dynamic programming algorithm. We encode a missing word by the negative logarithm of their probabilities to appear at the given locations and encode the mismatches by

calculating their semantic and morphology similarities. It requires fewer bits to encode between synonyms than antonyms.

Equation 4 must select the candidate question q, for which we use two strategies:

Offline. We cluster questions in Q and generate patterns offline, finding the most likely pattern, then generate q that is close to the input and to one of the patterns; and

Online. We consider only the question candidates relevant to the input and that could be matched by at least one of our templates generated from a few questions and that share some keywords with the input in Q.

Finally, we choose the best q that minimizes Equation 4. Furthermore, we apply a bigram language model to filter questions with low trustworthiness. The language model is trained in our background question set Q. The value δ is trained as a part of our experiments. In our system the δ value is a function of the lengths of the input questions.

We have thus implemented RSVP, which, given speech recognition input $\{q_1, q_2, q_3\}$, finds q such that Equation 4 is minimized.

Implementation Details

The following are some RSVP implementation details:

Step 1. Analyze input:

▸ Split the questions into words by Stanford Pos Tagger; at the same time, name entities are extracted from the input questions using Stanford NER, Linepipe NER, and YAGO; and

▸ Find the best alignments among the input questions through dynamic programming. Words or name entities with similar pronunciations are mapped together; for example, given three questions—whole is the mayor of Waterloo, hole is the mayor of Water, and whole was the mayor of Water—the best word alignment would look like this:

Whole is the mayor of Waterloo
Hole is the mayor of Water
Whole was the mayor of Water

Step 2. Improve input questions:

▸ Build a question based on the word-alignment results from Step 1; for each aligned word block, we choose one word to appear in the result;

▸ We assume that a good format question should contain a "wh"-word, including what, who, which, whose, whom, when, where, why, how, or what, or an auxiliary verb, including be, have, do, shall, will (would), shall (should), may (might), must, need, dare, or ought. If the inputs do not contain any such word, we add proper words into the question candidates; and

▸ Since some correct words may not appear in the input, we further expand the question candidates with homonym dictionaries and metaphone dictionaries.

Step 3. Analyze relevant database patterns:

▸ Find relevant database questions, sorting them based on their semantic and syntactic similarity to the improved input questions from Step 2;

▸ If a question is almost the same as one of the input questions, we return that input question directly, and no further steps are done in this case;

▸ The database questions involve many forms and patterns. We group similar questions together through a hierarchical clustering algorithm. The distance between two clusters is calculated based on the syntactic similarity between questions. The algorithm stops when the minimum distance between clusters reach a predefined threshold;

▸ When the relevant questions are grouped into clusters, we are able to extract patterns from each cluster. Following the algorithm outlined in Step 1, we align questions in each group, using their semantic similarities to encode the word distance. Then a group of questions is converted into a single list with multiple word blocks, with each block containing several alternative words from different questions; for example, given questions "Who is the mayor of New York," "Who is the president of United States," and "Which person is the leader of Toronto," we obtain a list of word blocks after alignment:

{who, who, which person}, {is}, {the}, { mayor, leader, president } of { New York, United States, Toronto}; and

▸ For each aligned word block, we further extract tags that would best describe the slot; here, YAGO[9] is used to describe the meaning of each word or phrase. We extract several most-common facts as the description of each word block. We then obtain one or several semantic patterns composed of words and facts from YAGO.

Step 4. Generate the candidate questions:

▸ Map the original input questions into the patterns we extracted from the database and replace the words in the patterns with the words from the input. Many candidate questions could be generated by considering the various combinations of word replacements; and

▸ To reduce complexity, we train a bigram language model from our question set, removing candidate questions with low probability.

Step 5. Rank candidate questions using information distance:

▸ Calculate the distance between the candidate questions and the input questions $K(q|I)$ and $K(I|q)$. We align the candidate and input questions and encode the word mismatches and missing words, encoding a missing word through minus logarithm of their probability to appear at the said

locations and calculating word mismatches through their semantic, morphology, and metaphone similarities;

▸ Calculate the distance between the candidate questions and the patterns $K(q|p)$. A method similar to the previous step is used to calculate the distances between questions and patterns; and

▸ RSVP Ranks all the candidates using Equation 4.

Step 6. Return the candidate with minimum information distance score as the final result: In order to improve speed, the last three items of step 3 may be performed offline on the complete database Q.

Completeness of the Database *Q*

We tested the hypothesis that Q contains almost all common question types. The test set T contained 300 questions, selected (with the criteria of no more than 11 words or 65 letters, one question in a sentence, and no non-English letters) from an independent Microsoft QA set at http://research.microsoft.com/en-us/downloads/88c0021c-328a-4148-a158-a42d7331c6cf. We found that all but three have corresponding patterns in Q. Only three questions lacked strictly similar patterns in Q: Why is some sand white, some brown, and some black? Do flying squirrels fly or do they just glide? And was there ever a movement to abolish the electoral college? We will provide the data set T upon request.

Experiments

Our experiments aimed to test RSVP's ability to correct speech-recognition errors in the QA domain, focusing on non-native speakers, as there are three non-native English speakers for each native English speaker in the world. Here, we further test and justify our proposed methodology by extending it to translation in the QA domain.

Experiment setup. We initially (in 2011) used the Nuance speech-recognition server and later switched to Google speech recognition (http://google.com), because the Google server has no daily quota and responds quicker. The RSVP system is implemented in a client-server architecture. The experiments were performed at a computer terminal with a microphone. The experimenter would read a question, and Google speech recognition would return three options. RSVP uses the three questions as input and computes the most likely question.

Dataset. We use the set T described in the previous section. T contains 300 questions. T was chosen independently, and $T \cap Q = \Phi$. Not all questions in T were used by each speaker in the experiments; non-native speakers and children skipped sentences that contain difficult-to-pronounce words, and less-proficient English speakers tend to skip more questions.

Time complexity. On a server with four cores, 2.8GHz per core, and 4G memory, RSVP typically uses approximately 500ms to correct one question; that is, the speaker reads a question into a microphone, Google voice recognition returns three questions, and RSVP uses the questions as input, taking approximately half a second to output one final question.

Human speaker volunteers. Such experiments are complex and time consuming. We tried our best to remove the individual speaker variance by having different people perform the experiments independently. We recruited 14 human volunteers, including native and non-native English speakers, adults and children, females and males (see Table 1) during the period 2011 to 2012.

We performed 12 sets of experiments involving 14 different speakers, all using the same test set T or a subset of T. Due to children's naturally short attention spans, the three native English-speaking children (two males, one female) completed one set of experiment (experiment 7), each responsible for 100 questions. A non-native-speaking female child, age 12, performed the test (experiment 10) independently but was able to finish only 57 questions.

In the following paragraphs, "CC" signifies that the speech-recognition software (from Google) returned the correct answer as the first option and RSVP agrees with it; "WC" signifies that the speech-recognition software returned the wrong answer as the first option and RSVP returned the correct answer; "CW" signifies that the speech-recognition software returned the correct answer as the first option and RSVP returned the wrong answer; and "WW" signifies that the speech-recognition software returned the wrong answer as the first option and RSVP also returned the wrong answer. All experiments were performed in quiet environments; in each, the speaker tried again if neither the speech recognition nor RSVP was correct (see Table 2).

Table 1. Individuals used in our experiments.

	Native Speaker		Non-Native Speaker	
	Adult	Child	Adult	Child
Female	0	1	4	1
Male	3	2	3	0

Table 2. Experimental results for speech correction.

Experiment	# questions	CC	WC	CW	WW	Base Translator Accuracy	RSVP Accuracy
Google as base translator	428	112	211	6	99	27.5%	75.6%
Microsoft as base translator	428	116	207	11	94	29.6%	75.6%
Google as base translator	52	21	15	0	16	40%	69%
Google as base translator	114	44	49	1	20	38%	81.6%

Table 3. Experimental results for speech correction

Experiment	Total No. of questions	CC	WC	CW	WW
1	164	105	39	5	15
2	300	219	25	6	50
3	300	222	15	5	58
4	257	141	41	7	68
5	181	100	26	4	51
6	214	125	29	10	50
7	206	145	19	8	34
8	298	180	12	4	102
9	131	77	14	0	40
10	57	28	4	1	24
11	63	35	9	1	18
12	107	62	9	2	34

Translation examples.

从多伦多到北京坐飞机多长时间
Google translation: Fly from Toronto to Beijing long?
Our translation: How long does it take to fly from Toronto to Beijing?
人身上有多少块骨头?
Google translation: People who have a few bones?
Our translation: How many bones do people have?
台湾的人口多少?
Google translation: Taiwan' s population size?
Our translation: What is the population size of Taiwan?
恐龙上什么时候灭绝的?
Google translation: When did the dinosaurs extinct?
Our translation: When did the dinosaurs become extinct?

Experiment 1. Non-native speaker, male. Here, the speaker chose only easy-to-read questions from the 300-question Microsoft test set; the following are examples where RSVP corrected Google's errors:

Original question: How many types of birds are found in North America? Google speech-recognition output: How many pints of birds are formed in North America? How many pints of birds are found in North America? How many pints of birds performed in North America? RSVP output: How many types of birds are found in North America?

Original question: How does the atom bomb work? Google speech-recognition output: call does the atom bomb work? All does the atom bomb work? aha does the atom bomb work? RSVP output: How does the atom bomb work?

Original question: How holidays are celebrated around the world? Google speech-recognition output: call holidays are celebrated around in the wall? call holidays are celebrated around in the world? how all holidays are celebrated around in the wall? RSVP output: how holidays are celebrated around in the world?

Original question: Are there any known aliens? Google speech-recognition output: are there any loans deviance? are there any loans aliens? are there any known deviance? RSVP output: Are there any known aliens?

Original question: What does a frog eat? Google speech-recognition output: what is front seat? what is the frogs eat? what does the front seat? RSVP output: What does frogs eat?

Experiment 2. Native speaker, male;

Experiment 3. Native speaker, male;

Experiment 4. Non-native speaker, male;

Experiment 5. Non-native speaker, female;

Experiment 6. Non-native speaker, female.

Experiment 7. Three native English-speaking children, 100 questions each. They are eight years old, female; nine years old, male; and 11 years old, male. In principle, we prefer independent tests with one individual responsible for the complete set of 300 questions. However, we were only able to get each of the children to do 100 questions, skipping the difficult ones. The result is similar to that of adult native English speakers;

Experiment 8. Native English speaker, male;

Experiment 9. Non-native English speaker, male;

Experiment 10. Non-native English speaker, female, 11 years old, in Canada to attend summer camp to learn English; her English was rudimentary and consequently was able to read only 57 questions out of 300;

Experiment 11. Non-native English speaker, female; and

Experiment 12. Non-native English speaker, female.

In our experiments, Table 2, the non-native speakers and the children selected relatively easy-to-read questions (without, say, difficult-to-pronounce names) from *T* to do the tests. The ratio of improvements was better for the non-native speakers, reducing the number of errors (WW column) by 30% on average for experiments 1, 4, 5, 6, 9, 10, 11, and 12. For native speakers, RSVP also delivered a clear advantage, reducing the number of errors (WW column) by 16% on average for experiments 2, 3, 7, and 8. Such an advantage would be amplified in a noisy real-life environment. Allowing the speaker to repeat the question would increase the success rate, as in the following example (with Google): RSVP generated "How many toes does Mary Monroe have?" for the first query and "How many titles does Marilyn Monroe have?" for the second query. Combining the two questions, RSVP generated the correct intended question "How many toes does Marilyn Monroe have?"

Translation

To further justify the methodology proposed here, we extend the approach to translation in the QA domain for cross-language search. Here, we use Chinese-English cross-language search as an example, though the methodology works for other languages, too.

A Chinese-speaking person can perform a cross-language search of the English Internet in two ways:

Translate it all. Translate the whole English Internet, including all QA pairs in the English QA community and English (Wikipidea) databases, into Chinese; or

Translate a question. Translate a Chinese question into English, finding the answer in English, then translate the answer back to Chinese.

General-purpose translators today perform so poorly that the first option is out of the question. The RSVP methodology enables the second option, which involves two translations: the Chinese question to English, then the English answer back to Chinese. Since the QA answers are usually simple and read by humans, and the database relations can be manually translated, a general-purpose translator is sufficient and sometimes not even needed. The key to this approach is translating Chinese questions into English. We implemented the translation system and cross-language (Chinese-English) search as part of the RSVP QA system through the following steps:

- Translate a Chinese question into English through a general-purpose translator;
- Apply the (modified) correction procedure described here;
- Perform English QA search; and
- Translate the result back into Chinese through a general-purpose translator.

Table 3 outlines experiments with our translation system, using the notation outlined earlier: CC, WC, CW, and WW. The first three used data collected as we developed RSVP; the fourth used an independent dataset of 114 questions on a range of topics (see the figure here).

Conclusion

This work would be more effective if it were integrated into speech-recognition software so more voice information could be used. However, it targets dynamic special domains that are so numerous that training them separately would be prohibitive.

In addition to special-domain translation, the RSVP methodology can be used to correct the grammatical errors and spelling mistakes in a normal QA text search, as well as to create an automatic writing assistant for a highly structured domain.

Moreover, "tuning" improves all systems; for example, if we ask "What is the distance between Toronto and Waterloo bla," then we know the extra "bla" should be removed, as inferred from system knowledge, requiring zero bits to encode such a deletion. The theory allows us to add "inferrable rules" at no additional cost.

Acknowledgments

We thank Li Deng of Microsoft for his advice on speech-recognition software and Leonid Levin of Boston University for discussions and suggestions on alternative formulations. We are grateful to the volunteers who participated in our experiments. We thank the referees and Nicole Keshav for helping us improve the article. We also thank Alan Baklor of Answer.com for his help. This work has been partially supported by Canada's IDRC Research Chair in Information Technology program, NSERC Discovery Grant OGP0046506, the Canada Research Chair program, a CFI Infrastructure grant, an NSERC Collaborative grant, Ontario's Premier's Discovery Award, and the Killam Prize. C

References

1. Baker, J.M., Deng, L., Glass, J., Khudanpur, S., Lee, C.H., Morgan, N., and O'Shaughnessy, D. Developments and directions in speech recognition and understanding, Part 1. *IEEE Signal Processing Magazine 26*, 3 (May 2009), 75–80.
2. Bennett, C.H., Gács, P., Li, M., Vitányi, P., and Zurek, W. Information distance. *IEEE Transactions on Information Theory 44*, 4 (July, 1998), 1407–1423.
3. Bennett, C.H., Li, M., and Ma, B. Chain letters and evolutionary histories. *Scientific American 288*, 6 (June 2003), 76–81.
4. Chen, X., Francia, B., Li, M., McKinnon, B., and Seker, A. Shared information and program plagiarism detection. *IEEE Transactions on Information Theory 50*, 7 (July 2004), 1545–1550.
5. Cilibrasi, R., Vitányi, P., and de Wolf, R. Algorithmic clustering of music based on string compression. *Computer Music Journal 28*, 4 (Winter 2004), 49–67.
6. Cilibrasi, R. and Vitányi, P. The Google similarity distance. *IEEE Transactions on Knowledge and Data Engineering 19*, 3 (Mar. 2007), 370–383.
7. Cuturi, M. and Vert, J.P. The context-tree kernel for strings. *Neural Networks 18*, 4 (Oct. 2005), 1111–1123.
8. Fagin, R. and Stockmeyer, L. Relaxing the triangle inequality in pattern matching. *International Journal of Computer Vision 28*, 3 (1998), 219–231.
9. Hoffart, J., Suchanek, F.M., Berberich, K., and Weikum, G. YAGO2: A spatially and temporally enhanced knowledgebase from Wikipedia. *Artificial Intelligence 194* (Jan. 2013), 28–61.
10. Keogh, E., Lonardi, S., and Ratanamahatana, C.A. Towards parameter-free data mining. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM Press, New York, 2004, 206–215.
11. Li, M. Information distance and its applications. *International Journal on the Foundations of Computer Science 18*, 4 (Aug. 2007), 669–681.
12. Li, M., Badger, J., Chen, X., Kwong, S., Kearney, P., and Zhang, H. An information-based sequence distance and its application to whole mitochondrial genome phylogeny. *Bioinformatics 17*, 2 (Feb. 2001), 149–154.
13. Li, M., Chen, X., Li, X., Ma, B., and Vitányi, P. The similarity metric. *IEEE Transactions on Information Theory 50*, 12 (Dec. 2004), 3250–3264.
14. Li, M. and Vitányi, P. *An Introduction to Kolmogorov Complexity and Its Applications, Third Edition.* Springer-Verlag, New York, 2008.
15. Lieberman, H., Faaborg, A., Daher, W., and Espinosa, J. How to wreck a nice beach you sing calm incense. In *Proceedings of the 10th International Conference on Intelligent User Interfaces* (2005), 278–280.
16. Lopes, L.R. A software agent for detecting and correcting speech recognition errors using a knowledge base. SATNAC, 2008.
17. Nykter, M., Price, N.D., Larjo, A., Aho, T., Kauffman, S.A., Yli-Harja, O., and Shmulevich, I. Critical networks exhibit maximal information diversity in structure-dynamics relationships. *Physical Review Letters 100*, 5 (Feb. 2008), 058702-706.
18. Nykter, M., Price, N.D., Aldana, M., Ramsey, S.A., Kauffman, S.A., Hood, L.E., Yli-Harja, O., and Shmulevich, I. Gene expression dynamics in the macrophage exhibit criticality. *Proceedings of the National Academy of Sciences 105*, 6 (Feb. 2008), 1897–1900.
19. Pao, H.K. and Case, J. Computing entropy for ortholog detection. In *Proceedings of the International Conference on Computational Intelligence* (Istanbul, Turkey, Dec. 17–19, 2004).
20. Rosenfeld, R. Two decades of statistical language modeling: Where do we go from here? *Proceedings of the IEEE 88*, 8 (Aug. 2000), 1270–1278.
21. Sarma, A. and Palmer, D.D. Context-based speech recognition error detection and correction. In *Proceedings of the Human Language Technology Conference* (Boston, May 2–7). Association of Computational Linguistics, Stroudsburg, PA, 2004, 85–88.
22. Veltkamp, R.C. Shape matching: Similarity measures and algorithms. In *Proceedings of the International Conference on Shape Modeling Applications* (Genoa, Italy, 2001), 188–197.
23. Zhang, X., Hao, Y., Zhu, X.Y., and Li, M. New information measure and its application in question-answering system. *Journal of Computer Science and Technology 23*, 4 (July 2008), 557–572.
24. Zhang, X., Hao, Y., Zhu, X., and Li, M. Information distance from a question to an answer. In *Proceedings of the 13th ACM SIGKDD Conference on Knowledge Discovery in Data Mining* (San Jose, CA, Aug. 12–15). ACM Press, New York, 2007, 874–883.

Yang Tang (tangyang9@gmail.com) is a research associate in the David R. Cheriton School of Computer Science at the University of Waterloo, Waterloo, Ontario.

Di Wang (kingwangdi@gmail.com) is a graduate student in the David R. Cheriton School of Computer Science at the University of Waterloo, Waterloo, Ontario.

Jing Bai (jbai@microsoft.com) is a researcher in Microsoft Corporation, Silicon Valley campus, Sunnyvale, CA.

Xiaoyan Zhu (zxy-dcs@tsinghua.edu.cn) is a professor in the Tsinghua National Laboratory for Information Science and Technology and Department of Computer Science and Technology and director of State Key Laboratory of Intelligent Technology and Systems at Tsinghua University, Beijing.

Ming Li (mli@uwaterloo.ca) is Canada Research Chair in Bioinformatics and a University Professor in the David R. Cheriton School of Computer Science at the University of Waterloo, Waterloo, Ontario.

© 2013 ACM 0001-0782/13/07