"Automatic Language Translation Generation Help Needs Badly"

Kevin Knight
Information Sciences Institute
University of Southern California

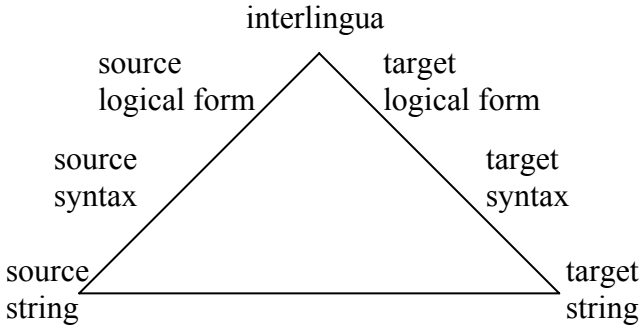
Machine translation systems output gobbledygook. This is not a pretty-printing problem. The solution requires a fundamental understanding of what makes a grammatical, sensible, and effective piece of text.

(Bad MT output is especially astounding when you consider that there are so many right answers!)

What's going wrong? For starters, we can look at human translation. My native language is English, and I can translate very well from Spanish to English, but not in reverse, from English to Spanish. I can comprehend the meaning of both Spanish and English sentences just fine, but I cannot generate a good Spanish sentence, because I don't know Spanish well enough. We see an analogous situation in MT. There are analysis problems, but the main problem is generation.

Inside the Pyramid

The MT pyramid looks like this:



As generation researchers, we could just stand at the top and say: "We're ready for our input." There are problems with this approach:

(First) It doesn't take us anywhere.

(Second) We're definitely not ready. Imagine: here comes the input, expressed in some formalism whose BNF includes a conceptual inventory as well as where the parentheses go. We certainly don't have the tools and knowledge resources today to generate from that input, to say nothing of measuring generation accuracy. Say we want to be ready by the year 2020. What should we be doing today? Analysis researchers are quite interested in Ontobank and related semantic annotation efforts. MT researchers have also stuck their noses into this business and said to Ontobank: you are annotating English, Chinese, and Arabic texts, but it's vital to annotate *parallel* texts, so we can see how the semantic structures match up. Likewise, it's important to make sure that this semantic annotation effort creates resources and opportunities for generation research.

(Third) We're getting generation input already! It's coming from people who are tunneling through the lower levels of pyramid. These are the same people producing the above-mentioned gobbledygook. We're tempted to say, "Bad output? You deserve it, for tunneling through the pyramid." But maybe we can help.

Transformation formalisms

When we map from one thing to another (e.g., meaning to text), we are trying to capture a mathematical relation, which is a possibly infinite set of pairs (e.g., pairs of meanings and sentences). There are benefits to thinking about capturing that set of pairs in a nice formalism. For example, we might be able to get inference algorithms off the shelf. ("Efficiently give me the 10,000-best outputs for this one input"). As another example, we might get reversibility for free. ("Give me all the inputs that would generate this output"). Many generation systems have declarative grammar representations for mapping meanings into sentences, and they do implicitly capture a mathematical relation, but they are often hard to reverse. The early statistical generation systems could generate many English sentences for a given meaning, but they could not take an English sentence and generate all the meanings that (if put through the system) would output that sentence. If we could do that, then we could easily imagine building a useful English-to-English paraphraser by composing an English-to-meaning transformer with a meaning-to-English transformer.

By contrast, statistical MT systems are routinely built on reversible formalisms, whether based on phrase substitution/transposition, synchronous grammars, or tree transducers. For example, the transducer rules that power a recent foreign-string-to-English-tree MT system [Galley et al 04] have been used to build a reverse English-tree-to-foreign-string MT system [Huang, Joshi & Knight 06]. Moreover, phrase substitution systems have recently been strung together to build the paraphrasing capability mentioned above: instead of composing an English-to-meaning transformer with a meaning-to-English transformer, they compose an English-to-French transformer with a French-to-English transformer [Bannard & Callison-Burch 05].

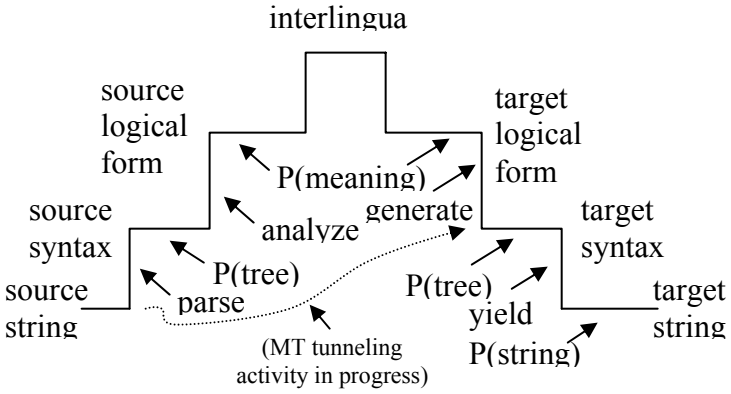
Recently, [Wong & Mooney 06] have been applying synchronous grammars to analyzing sentence meanings in database-query domains, and fortunately, they also has explored using that formalism for the reverse problem of generation [Wong & Mooney 07]. Since these systems operate in several languages, we may start to see compositions of these tools aimed at doing meaning-based MT. Wong & Mooney have also exploited another benefit of these formalisms, which is the ability to train from input/output samples.

Language models

The process of over-generating and scoring outputs is common in statistical MT. There, a target-language n-gram model pulls a lot of weight. With its specialized knowledge about the target language, it helps craft fluent text and relieves the translation system from having to know everything about the source language, and from having to know everything about how to transform source to target. The language model also does lexical selection and word ordering. This is certainly the province of generation.

Typical language models operate at the string level, i.e., they compute P(string) using n-gram frequencies. Future language models will operate at all levels of the MT pyramid, not just at the string level. For ten years already, parsing researchers have been building P(tree) models, and these have still not entered the mainstream of MT generation. We will also have P(meaning) and P(interlingua) models in the future. These will lift a heavy burden off of transformation systems, which will then be mainly responsible for generating lots of outputs for any given input.

When we add the language models, we can see that we've really got an MT ziggurat on our hands:



Software Tools

What tools do MT researchers use when building their systems? Here are some:

1. SRI-LM, CMU-CU: tools for building n-gram language models
2. Charniak/Collins/Stanford parsers: tools for parsing bilingual MT training data
3. BBN Serif, etc.: tools for finding names in MT data
4. AT&T FSM: tools for building and composing weighted finite-state transducers

Some of these are strictly algorithms (1 and 4), while others are algorithms shipped with knowledge resources (2 and 3). There are no recognizable generation tools in this list. There are a couple of reasons for this. First, there are not that many easily installable, easily runnable generation tools. Second, MT systems use an integrated search, in which all knowledge resources are brought to bear simultaneously on the input sentence. This makes it hard to tack an existing software tool on to the end of the process. (An exception is the ubiquitous *re-capitalization* tool, needed because many base MT systems output lower-case text). Tool pipelining can be effective when the pipes are thick -- so, generation tools may need to accept packed input representations that store many inputs, all of which are to be processed.

What are some potential generation tools? Every week, some researcher somewhere needs an English morphological generator. As MT research moves into more inflected target languages, morphological generation will continue to mount as a serious need.

To any extent that meaning-to-text tools can lift the burden of producing target language, they will also be studied carefully. To put it crudely, MT systems work better if their inputs and outputs are more "normalized."

It's also important that generation tools come packed with knowledge; by analogy, a bottom-up chart parsing algorithm is not nearly as useful as the Charniak parser.

Even if generation tools are not welded directly into MT systems at first, the knowledge inside of them is critical, and this knowledge will certainly be added in. For example, a tool that can correctly order a bag of English words into a grammatical, sensible string will be of immense interest. Initial results can be seen in [Soricut & Marcu 05]; in this work, we can also see input representations that pack many individual inputs.

Tools get evaluated, and this helps drive progress. There are tools and evaluations for morphological analysis (how about morphological generation), parsing and semantic role

labeling (how about meaning-to-text), co-reference detection (how about reference generation), word sense disambiguation (how about lexical selection). It's surely coming.

... and beyond?

Yorick Wilks once joked that generation is harder than analysis, because analysis is like counting from one up to infinity, while generation is like counting from infinity down to one. Well, if the MT triangle is *infinitely* high, then okay. But maybe generation is more like counting from 9 billion down to one. Or maybe (even better), this is the right way to count:

1
2, 1
3, 2, 1
4, 3, 2, 1
5, 4, 3, 2, 1
…

And the additional saving grace -- when we generate language, there are so many right answers!

References

[Bannard & Callison-Burch 05] "Paraphrasing with Bilingual Parallel Corpora", ACL.

[Galley, Hopkins, Knight, Marcu 04] "What's in a Translation Rule?" NAACL-HLT.

[Huang, Knight & Joshi 06] "Statistical Syntax-Directed Translation with Extended Domain of Locality", AMTA.

[Soricut & Marcu 05] "Towards Developing Generation Algorithms for Text-to-Text Applications", ACL.

[Wong & Mooney 06] "Learning for Semantic Parsing Using Statistical Machine Translation Techniques", NAACL-HLT.

[Wong & Mooney 07] "Generating by Inverting a Semantic Parser that Uses Statistical Machine Translation", NAACL-HLT.