CSME 206A Natural Language & Speech Processing **Spring Semester**

Lecture 11: NLP For Devnagri

Lecturer: K.R. Chowdhary *: Professor of CS*

11.1 Introduction

He Complexity and diversity of human languages makes automated translation one of the hardest problems in computer science. Yet the job is becoming more important as writing and speech are increasingly digitized and as the traditional separations between societies dissolve.

Few parts of the globe have as much need to translate from one language to another as does India. According to India's 2001 census, the country has 122 languages, 22 of which are designated as official languages by the government. The top six – Hindi, Bengali, Telugu, Marathi, Tamil, and Urdu – are spoken by 850 million people worldwide.

Now a decades-long effort by researchers is about to bear fruit. A multipart machine translation architecture, Sampark, is nearing completion as the combined effort of 11 institutions led by the Language Technologies Research Center at the International In- stitute of Information Technology in Hyderabad (IIIT-H).

Sampark combines both traditional rules- and dictionary-based algorithms with statistical machine learning, and will be rolled out to the public at http:// sampark.iiit.ac.in/.

Many Indian languages are derived from Sanskrit, which is based on rules set down by Panini, the 4th century B.C. grammarian. Even those Indian languages that are not derived from Sanskrit are structurally similar to others in India. This common underpinning makes the translation from one Indian language to another easier than from, say, German to Chinese. Nevertheless, there are 462 pair-wise translations (counting each direction for a pair) possible among the 22 official Indian languages, so clearly the researchers had to find a generalized approach that could be easily adapted from one language to another.

The chosen method, a transfer-based approach, consists of three major parts: analyze, transfer, and generate. First, the source sentence is analyzed, then the results are transferred in a standard format to a set of modules that turn it into the target language. Each step consists of multiple translation "modules."

An advantage of the three-step approach, is that a particular language analyzer, one for Telugu, for example, can be developed once, independent of other languages, and then paired with generators in various other languages, such as Hindi [gathens10].

The 13 major translation modules together form a hybrid system that combines rules-based approaches – where grammar and usage conventions are codified – with statistical-based methods in which the software in essence discovers its own rules through "training" on text tagged in various ways by human language experts, as shown in Fig. 11.1.

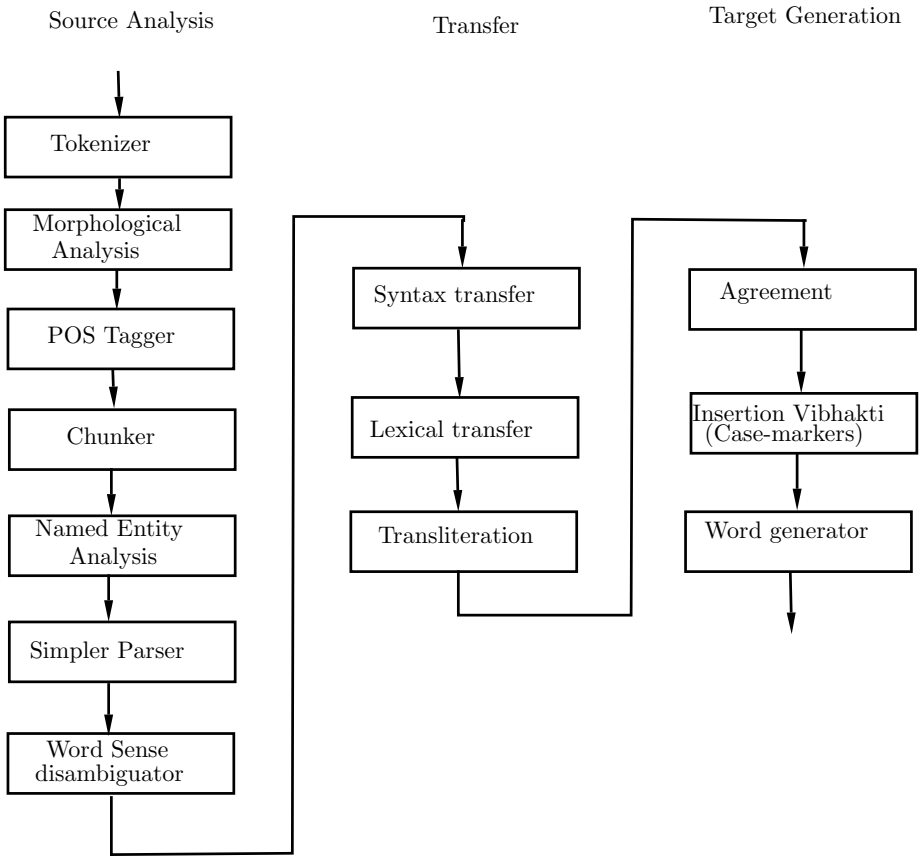

Figure 11.1: Working of 'Smpark' module

11.2 A Transfer-Based Approach

Translation systems for major languages today – from companies like Google and Microsoft, for example – often use statistical approaches based on parallel corpora, huge databases of corresponding sentences in two languages.

These systems use probability and statistics to learn by example which translation of a word or phrase is most likely correct. And they move directly from source language to target language with no intermediate transfer step.

"The statistical direct translation approach is, in a sense, the lazy man's approach, because all it requires is that you go and hunt for parallel corpora and you turn the crank and you get what you get. But the transfer-based approach is much more linguistically motivated, because you are trying to analyze the sentence and trying to arrive at something that is close to a representation of its meaning.

Parallel corpora are specialized databases consisting of sentences very carefully translated and then mapped one-for-one to their translations. More over, to do a good job of training translation systems, the parallel corpora must be very large –in the billions of sentences. "People are coming to grips with the fact that parallel data are not easy to come by. This is a very specialized kind of data. Indeed, parallel corpora for many Indian language pairs do not exist and cannot easily be built, in part because not much Indian language text has been digitized.

Nevertheless, developers at the Language Technologies Research Center were able to apply statistical machine learning in a limited way by annotating small monolingual corpora and analyzing the tagged text with statistical techniques. So although machine learning techniques were employed in some of the modules, developers painstakingly developed multilanguage dictionaries and codified rules in the Computational Paninian Grammar framework. They also held workshops of experts of all these languages to develop a standard tag set, and then used those tags to an- notate the monolingual corpora.

In fact most machine translation is not inspiration, but the perspiration. The hard part is building all the resources required, like dictionaries, morphological analyzers, parsers, and generators. It's a lot of grunt work. The effort that Sampark developers put into language analysis could have a broad impact beyond translating Indian languages. Even the best purely statistical systems can be made more accurate by first doing the types of detailed language analysis employed in Sampark. What one can do in the future is to first do monolingual analysis of one or both sides in paralleled corpora, and then use that to improve the quality of machine learning from the parallel corpora. So what we have done would also be useful if larger parallel corpora became available in future.

Another advantage of the transfer approach, is its generalizability. If you give me a parallel corpus dealing with financial news, and I train it up with millions of sentences of that sort, and two days later you say, 'Translate a sports article,' it's not going to perform as well." But that kind of application domain change has been explicitly anticipated by Sampark's developers. The first version, rolling out now language-by-language, is general purpose and optimized for tourism-related uses, but it will be made available to large users who wish to customize it for other domains, says Dipti Sharma, an associate professor at IIIT-H. That would involve building a new domain dictionary, incorporating rules that handle domain-specific grammatical structures, and perhaps retraining some modules such as Part of Speech Tagger and Named Entity Recognizer.

The effort required to make those changes is minimized by building on the existing multilingual dictionary, Sharma says. It is sense- or meaning-based, so that for one domain or language, "bank," for example, would most likely represent a financial institution, but for another it might refer to the edge of a river, Sharma says. The dictionary currently allows translation among nine languages.

The language-translation system has two especially noteworthy attributes. First, the linguistic analysis based on Panini is "extremely good," he says. It was initially chosen for Indian languages, but we find it is also suitable for other languages. Initially, hard work is needed, he says, in setting it up by developing standards for parts-of-speech tags and dependency tree labels and for figuring out ways to handle unique language constructs.

The second attribute of special note is the system's software architecture. It is an open architecture in which all modules produce output in Shakti Standard Format (SSF). The architecture allows modules written in different programming languages to be plugged in. Readability of SSF helps in development and debugging because the input and output of any module can be easily seen. Also, a dashboard tool supports the architecture in a variety of ways. Custom written, it is "extremely robust," Sangal says. "If a module fails to perform a proper analysis, the next module will still work, albeit in a degraded mode. So the system never gives up; it always tries to produce something."

An automated system for translating one Indian language to another, sampark (Fig. 11.1), is a hybrid system consisting of traditional rules-based algorithms and dictionaries and newer statistical machine-learning techniques. it consists of three major parts and 13 modules arranged in a pipeline. Following is the explanation of various blocks of Sampark.

Source Analysis:

- *Tokenizer:* Converts text into a sequence of tokens (words, punctuation marks, etc.) in shakti standard

Format.

- *Morphological analyzer:* Uses rules to identify the root and grammatical features of a word. splits the word into its root and grammatical suffixes.
- *Part of speech tagger:* Based on statistical techniques, assigns a part of speech, such as noun, verb or adjective, to each word.
- *Chunker:* Uses statistical methods to identify and tag parts of a sentence, such as noun phrases, verb groups, and adjectival phrases, and a rule base to give it a suitable chunk tag.
- *Named entity recognizer:* Identifies and tags entities such as names of persons and organizations.
- *Simple parser:* Identifies and names relations between a verb and its participants in the sentence, based on the Computational paninian Grammar framework.
- *Word sense disambiguation:* Identifies the correct sense of a word, such as whether "bank" refers to a financial institution or a part of a river.

Transfer:

- *Syntax transfer:* Converts the parse structure in the source language to the structure in the target language that gives the correct word order, as well as a change in structure, if any.
- *Lexical transfer:* Root words identified by the morphological analyzer are looked up in a bilingual dictionary for the target language equivalent.
- *Transliteration:* Allows a source word to be rendered in the script of the target language. Useful in cases where translation fails for a word or a chunk.

Target Generation:

- Performs gender-number-person agreement between related words in the target sentence.
- *Insertion of Vibhakti:* Adds post position and other markers that indicate the meanings of words in the sentence.
- *Word generator:* Takes root words and their associated grammatical features, generates the appropriate suffixes and concatenates them. Combines the generated words into a sentence.

References

[1] Manoj kumar Sharma and Debasis Samanta, *Word Prediction System for Text Entry in Hindi*, *ACM Transactions on Asian Language Information Processing*, Vol. 13, No. 2, Article 8, Publication date: June 2014.

[2] Gary Anthes, *Automated Translation of Indian languages*, DOI:10.1145/1629175.1629184, communications of the ACM, Jan. 2010 — Vol. 53, No.1, pp. 24-26