GF and Machine Translation

Aarne Ranta

Computational Syntax course, Gothenburg, 2014

Plan

Machine translation background

GF's formal potential as a translation system

Domain-specific vs. open-domain translation

Open-domain problems: interlingual lexicon, robustness, disambiguation

Probabilistic GF grammars

Learning GF grammars from data

The current GF translation systems: web and mobile

Machine translation background

Background overview

Some history of MT

Methods: rule-based, statistical, hybrid; interlingua, transfer

Evaluating MT

What is easy and what is difficult

A prediction

Five, perhaps three years hence, interlingual meaning conversion by electronic process in important functional areas of several languages may well be an accomplished fact.

A prediction

Five, perhaps three years hence, interlingual meaning conversion by electronic process in important functional areas of several languages may well be an accomplished fact.

IBM press release 1954

http://www-03.ibm.com/ibm/history/exhibits/701/701_translator.html

Early history

Turing: one of the things a machine could do

Shannon, Weaver: cryptography

- Russian is encoded English

optimism

The first critiques

Bar-Hillel (1960): *the pen is in the box*

The ALPAC report (1966): MT is low quality, useless, too expensive

Kay: MT must be interactive

Knowledge-based systems

Systran: transfer rules

Meteo: domain-specific (weather reports)

Rosetta: interlingual (Montague grammar)

VerbMobil: speech translation (unification grammar, Prolog)

The return of statistics

IBM: French to English trained at the Hansards corpus of Canadian Parliament

Google translate: on-line, 80 languages, based on the IBM ideas

Bing: Microsoft's on-line translator

Giza++ and Moses: open-source software for statistical MT

Pendulum swung too far?

Church (2011): there's no more low-hanging fruit

Hybrid systems: find the best combination of linguistics and statistics

Apertium: rule-based translation for closely related languages

GF: interlingual translation based on shared semantics

Rule-based methods

Word to word (dictionary lookup)

Rearrangement (of words)

Structure to structure (hierarchic phrases, not just words)

Use of grammars: morphology, syntax, semantics

Statistical methods

Noisy channel: French is distorted English

Word alignment: find corresponding words by looking at parallel texts

Language model: n-grams (sequences of *n* words)

Phrase-based: from words to multiwords (*for example*, *in spite of*)

Training: building the model from data

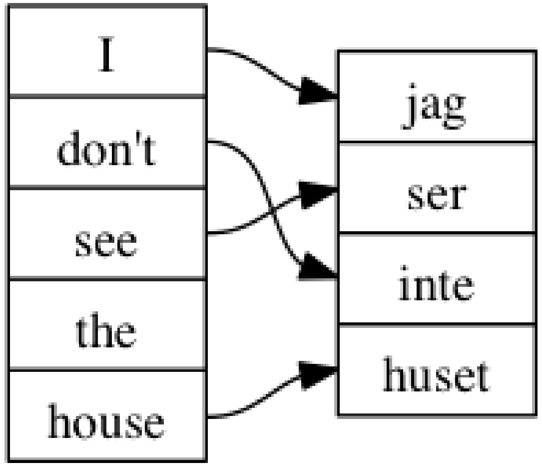
Decoding: applying the model at run time

The noisy channel formula

$\hat{e} = argmax\ P(f|e)P(e)$

- $P(f|e)$ **translation model**: probability of e being distorted to f
- $P(e)$ **language model**: probability of e in target language

Word alignment



Choice of alignment

house	*hus*, *huset*
houses	*hus*, *husen*
is	*är*
are	*är*
am	*är*
red	*röd*, *rött*, *röda*
this	*det här*, *det där*, *denna*, *detta*

this house is red:

den här hus är röda? *det här huset är rött*?

Language model: n-grams

n-gram = sequence of *n* words ($n = 1,2,3,4,...$)

3-grams in Swedish:

- **den här hus*
- *det här huset*
- **huset är röd*
- *huset är rött*

det här huset + *huset är rött* -> *det här huset är rött*

Phrase alignment

Alignments of common multiwords

Helps with idiomacy

vice president

-> *vice ordförande* (Google translate)

-> *skruvstädspresident* (GF baseline translator)

Hybrid methods

Language = structures + distribution

Don't guess if you know

Factored systems: from words to lemma+analysis pairs

Tree-based systems: probabilistic grammars

Transfer vs. interlingua

Transfer: rules for each language pair

Interlingua: use an intermediate language

- a pivot language (English, Esperanto)
- a meaning representation (formal logic)

For n languages, interlingua needs $2n$ components, transfer needs $n(n\text{-}1)$

Sharing effort: perform operations on interlingua level

Linked Wordnets as interlingua: 80% of words in one-to-one correspondance

Evaluating MT: manual evaluation

Quality criteria

- grammaticality
- fidelity (meaning preservation)
- fluency

Measure

- post-editing effort
- edit distance

Evaluating MT: automatic evaluation

Gold standard: typically a separate part of the training material

Word error rate: how many words don't match with gold standard

BLEU: match words and n-grams (sequences of n words)

Evaluation as training: set parameters to maximize the BLEU score

BLEU

Geometric mean of 1-gram, 2-gram, 3-gram, and 4-gram precisions multiplied by a brevity penalty

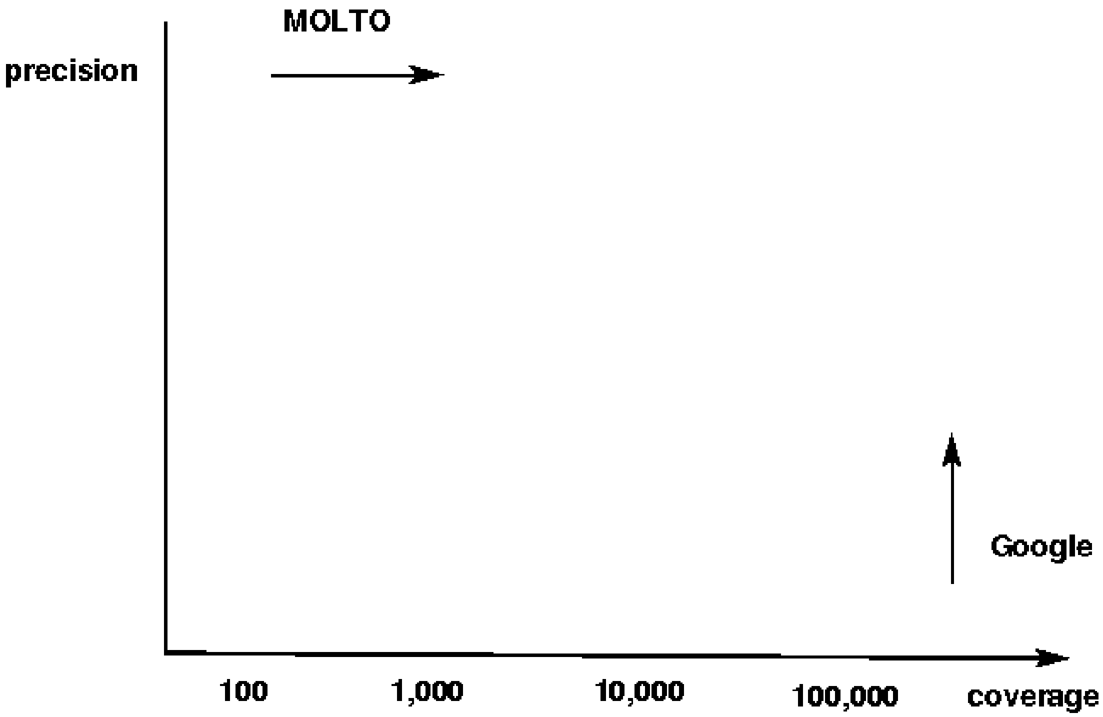
Trade-offs in translation

Coverage vs. **precision**

Browsing vs. **publication** (a.k.a **assimilation** vs. **dissemination**)

Bar-Hillel (1960): you cannot achieve both coverage and precision at the same time

Two systems and their ambitions



Coverage

Estimate of information needed: 100k words, 100M 2-grams, 10G 3-grams

Morphological variation: 1000k word forms, 1000M 2-grams, 1000G 3-grams

Sparseness of data: hard to find all this

Smoothing: if you cannot find the 3-gram, combine two 2-grams

- *is here is* = *is here* + *here is*

The Vauquois triangle

```
           semantics
              /\
             /  \
            /    \
           /      \
          /        \
         /          \
        /            \
       /              \
lang1 /________________\ lang2
```

Levels of analysis

there are five gay bars in this street

```
           semantics
              /\
             /  \
            /    \
           /      \
          /        \
         /          \
        /            \   där är fem glad stänger i detta gata
       /              \
  Eng /________________\ Swe
```

Levels of analysis

there are five gay bars in this street

```
          semantics
             /\
            /  \
           /    \
          /      \
         /        \     där är (fem glada stänger) i (denna gata)
        /          \
       /            \   där är fem glad stänger i detta gata
      /              \
 Eng /________________\ Swe
```

Levels of analysis

there are five gay bars in this street

```
         semantics
            /\
           /  \
          /    \       det finns fem glada stänger på denna gata
         /      \
        /        \     där är fem glada stänger i denna gata
       /          \
      /            \   där är fem glad stänger i detta gata
     /              \
Eng /________________\ Swe
```

Levels of analysis

there are five gay bars in this street

```
         semantics
            /\         det finns fem gay-barer på denna gata
           /  \
          /    \       det finns fem glada stänger på denna gata
         /      \
        /        \     där är fem glada stänger i denna gata
       /          \
      /            \   där är fem glad stänger i detta gata
     /              \
Eng /________________\ Swe
```

Long-distance dependencies

Agreement (French):

- *my father is intelligent - mon père est intelligent*
- *my mother is intelligent - ma mère est intelligent***e**
- *my mother is actually, regardless of what you say, very intelligent*

Discontinuous verbs (German):

- *er* **bringt** *dich* **um** - *he* **kills** *you*
- *er* **bringt** *deinen besten Freund* **um** - *he kills your best friend*

Reordering

The snow is white. If the snow is white, then the snow is white.

German: three orders,

Der Schnee ist weiss. Wenn der Schnee weiss ist, dann ist der Schnee weiss.

Disambiguation

I sent four **letters** *to the president*

I ate a pizza with shrimps
I ate a pizza with friends
I ate a pizza with chopsticks

Pros and cons of RBMT and SMT

Not just precision vs. coverage:

Grammatical correctness: RBMT

Meaning preservation: ?

Reordering: RBMT

Long distance; RBMT

Disambiguation: SMT?

Fluency: ?

Idioms, multiwords: SMT

Low-resourced languages: RBMT?

Effort needed: SMT

Predictability: RBMT

Programmability: RBMT

Ideal languages for SMT

Morphologically simple

Rigid word order

Lots of data

English! And Swedish, Dutch, French,...

Ideal languages for RBMT?

Morphologically complex

Free/varying word order

Lack of digital data

Notoriously bad for SMT: Finnish, Japanese,...

An ideal hybrid system?

Taking all pros and cons into account

Easy to say, not so easy to do

GF

Multilingual grammar formalism based on type theory and functional programming

Multilingual grammar = abstract syntax + concrete syntaxes

Parsing: from string to abstract syntax

Linearization: from abstract syntax to string

Translation = parsing followed by linearization

Abstract syntax is interlingua

Potential

GF uses PMCFG = Parallel Multiple Context-Free Grammar

- between context-free and context-sensitive; slightly stronger than TAG (Tree-Adjoining Grammar)

Efficient runtime (empirically linear parsing)

Probabilistic GF grammars (abstract syntax probabilities)

Robust parsing: recovery from out-of-grammar parts of input

Synchronous grammars

Synchronous CFG: two rhs's (e.g. English and Latin)

```
S  -> NP VP   | NP VP
VP -> V2 NP   | NP V2
V2 -> "loves" | "amat"
```

Synchronous grammars

Synchronous CFG: two rhs's (e.g. English and Latin)

```
S  -> NP VP   | NP VP
VP -> V2 NP   | NP V2
V2 -> "loves" | "amat"
```

Synchronous PMCFG: English and Dutch

```
S  -> NP VP   | NP VP
VP -> V2 NP   | V2.1 NP V2.2
V2 -> "loves" | <"heeft","lief">
```

Multilingual GF grammars

Synchronous PMCFG

- generalization of synchronous CFG
- different lincat's, discontinuous constituents
- this enables a common abstract syntax in ”almost all cases”
- works for all languages so far (29 in the Resource Grammar Library)

Moreover: high-level source language for grammar engineering

RGL, the Resource Grammar Library

Implemented for 29 languages

Afrikaans	Bulgarian	Catalan	Chinese	Danish
Dutch	English	Estonian	Finnish	French
German	Greek	Hindi	Italian	Japanese
Latvian	Maltese	Nepali	Norwegian	Persian
Punjabi	Polish	Romanian	Russian	Sindhi
Spanish	Swedish	Thai	Urdu	

In progress: Arabic, Hebrew, Turkish, ...

Some RGL statistics

50+ contributors 2001-

3-6 months for a new language

3000-5000 lines of GF code per language

Complete morphology engine, comprehensive syntax, test lexicon (500 lemmas)

Larger dictionaries (10k - 100k lemmas) for 13 languages

Translation lexicon availability

https://docs.google.com/spreadsheets/d/1NuLRp86UPjd298LxjhCAGIHsoF

11 languages

16k to 66k lemmas, mostly extracted automatically

a few hundreds or thousands of checked lemmas

Translation systems of different types

Application grammars

- interlingua based on domain semantics: `Like x y`
- RGL used as library: `Like x y = mkCl x like_V2 y`
- compile-time transfer: `Like x y = mkCl y piacere_V2 x`
- limited but high quality

Resource grammars

- interlingua based on syntactic structures in the RGL
- structure-to-structure translation
- open-ended, but not full quality

Problems solved in application grammars

Translation via semantics (the top of the Vauquois triangle)

Choice of proper idiom: *please* to *s'il vous plaît*, *bitte*,...

Ambiguity reduced: *bank* may only mean the financial institution

Transfer of syntactic structure: *I like this* to *questo mi piace*

Current status in GF-based translation

Application grammars dominate: mathematics, painting descriptions, tourist phrasebook, Attempto controlled language, dialogue systems, pharmaceutical patents, software specifications, contracts, ...

RGL is used as a library, to make application grammar building easy

- less effort than manual coding (by orders of magnitude)
- no linguistic knowledge required from domain experts

A typical application has 15 languages and 200-500 concepts (i.e. abstract syntax functions)

Use cases

Production/publication/dissemination quality can be reached by automatic translation

Broadcasting to many languages

Web interfaces and mobile device app's (Android, iPhone)

Predictive parsing and syntax editing guide the user to enter translatable input

But this is not what mainstream machine translation does!

Translating uncontrolled input: simple idea

Resource grammar syntax + large dictionary

"Syntactic transfer" in Vauquois triangle

Refinements

Statistical disambiguation

Robust parsing

Back-up strategies

Controlled language core

"all levels of the Vauquois triangle in one system"

Multilingual lexicon

The first problem to solve: what is the abstract syntax

Words don't match one-to-one between languages

So, what is the abstract syntax?

Thinking of semantics: it is **word senses**

Thus different fun's for letter (character) and letter (document)

The Princeton WordNet

A lexical database for English words: http://wordnet.princeton.edu/

Words may have different senses

The senses are organized in hierarchies: **synonyms**, **hypernyms**, etc

Synset: set of synonymous words, i.e. a word sense

The 3.0 database contains 155,287 words organized in 117,659 synsets for a total of 206,941 word-sense pairs (http://en.wikipedia.org/wiki/WordNe

Linked WordNets

WordNets for other languages, mapping their words to Princeton senses

Rather complete ones:

- Finnish http://www.ling.helsinki.fi/en/lt/research/finnwordnet/
- Hindi http://www.cfilt.iitb.ac.in/wordnet/webhwn/

Many incomplete, automatically extracted ones: Universal Wordnet http://www.mpi-inf.mpg.de/yago-naga/uwn/

General observation: 80% of mappings are unproblematic

But a bit too fine-grained, and English-directed

Multilingual GF dictionary

Start with English words as abstract syntax id's

Split senses if needed in other languages

Expect this to converge

Variants, with the most frequent synonym first

If a synonym doesn't exist, use a hypernym

- *octopus, squid, cuttlefish* -> *bläckfisk* ("cephalopod")

Convergence of word sense splitting

English	Swedish	French
time		

Convergence of word sense splitting

English	Swedish	French
time	*gång*	
,,	*tid*	

Convergence of word sense splitting

English	**Swedish**	**French**
time	*gång*	*fois*
,,	*tid*	*temps*

Convergence of word sense splitting

English	**Swedish**	**French**
time	*gång*	*fois*
,,	*tid*	*temps*
weather	*väder*	,,

Robustness by metavariables

If parsing fails, e.g. with unknown words, the parser tries to fill in a **metavariable** (placeholder, unknown subtree)

```
p "he ate a ftira"
UseCl Past (Pred he_NP (Compl eat_V2 ?))
```

The easiest way to solve this is to return the original word in translation

```
er ass ein ftira
```

(Related case: if there's no German linearization, return the English word)

Metavariables can also occur in nodes that the construction doesn't parse:

```
p "her he loves"
UseCl Present (? she_NP he_NP love_V2)
```

There's no complete theory about how to handle these yet.

Robustness by chunking

Alternative to metavariables

+ more fine-grained

+ faster

- manual work for each language

Surprisingly easy on top of the RGL

Inspired by Apertium - and the MT of the 1950's

Disambiguation, the problem

Different senses of a word may translate to different words

- *this number is even -> diese Zahl ist gerade*
- *this surface is even -> diese Fläche ist eben*

Different syntactic structures may have different linearizations

- *I ate (a pizza with shrimps) -> j'ai mange une pizza aux crevettes*
- *I (ate a pizza with shrimps) -> j'ai mange une pizza avec des amis*

Word-sense disambiguation, grammar-based

A simple solution, using fine categorization

```
cat
  Number ;
  Surface ;
  Proposition ;
fun
  EvenNum : Number -> Proposition ;
  EvenSur : Surface -> Proposition ;
```

A more powerful solution, using **dependent types** to express **selectional restrictions**:

```
cat
  Class ;
  Term (c : Class) ;
  Property (c : Class) ;
  Proposition ;
fun
  Number, Surface : Class ;
  Pred : (c : Class) -> Term c -> Property c -> Proposition ;
  EvenNum : Property Number ;
  EvenSur : Property Surface ;
```

Word-sense disambiguation, statistical

Target language n-grams with wrong senses of words are rare.

Test this in Google translate!

Also, what happens when the distance gets larger.

(Also test syntactic disambiguation with prepositions.)

Syntax disambiguation

Syntactic parsing easily gives thousands of trees

Statistical disambiguation: rank with **tree probabilities**

Estimate probabilities from **treebanks**

Penn Treebank: http://www.cis.upenn.edu/~treebank/

- a set of 40k manually parsed sentences from Wall Street Journal
- converted to GF RGL abstract trees (Angelov 2012)

Tree probability in CFG

Probabilistic CFG: each rule has a probability

```
S  -> NP VP   -- 0.9
VP -> V2 NP   -- 0.3
NP -> "John"  -- 0.1
NP -> "beer"  -- 0.1
V2 -> "likes" -- 0.1
```

Sentence probability = tree probability = product of rule probabilities

```
p(John likes beer) = p((s (NP john) (VP (V2 likes) (NP beer))))
                   = 0.9 * 0.1 * 0.3 * 0.1 * 0.1 = 0.00027
```

Tree probability in GF

Probabilistic GF: each abstract syntax function has a probability

```
Pred  : NP -> VP -> S    -- 0.9
Compl : V2 -> NP -> VP   -- 0.3
John  : NP               -- 0.1
Beer  : NP               -- 0.1
Like  : V2               -- 0.1
```

Works the same way as probabilistic CFG:

```
p(John likes beer) = p(Pred John (Compl Like Beer))
                   = 0.9 * 0.1 * 0.3 * 0.1 * 0.1 = 0.00027
```

How to estimate probabilities

Relative frequencies of nodes in treebanks (sum to 1 per category)

But: there are no treebanks for many languages

In GF, one can port tree probabilities from one language to another, if the abstract trees are shared! (This is of course just an approximation. It should work better for semantics than for fluency.)

Advantage over PCFG: more abstract trees -> less sparse data

A problem

p(*John likes beer*) = p(*beer likes John*)

This is because the probabilities are context-free.

Could be improved by

- using "n-grams of tree nodes"
- dependent type probabilities (a research topic)

Learning GF grammars from data

Idea:

1. use human translator or SMT as source
2. parse with resource grammar
3. recognize a **construction** (a frequent abstract tree pattern)
4. introduce a special rule for it
5. recognize the same construction in parallel data

This generalizes the recognition of phrases in phrase-based SMT

Abstracting out a construction

Data:

John is five years old	`Pred John (ComplAP (Mod (Num 5 Year) Old))`
they are seventy years old	`Pred They (ComplAP (Mod (Num 70 Year) Old))`
I am fifteen years old	`Pred I (ComplAP (Mod (Num 15 Year) Old))`
repeated pattern:	`Pred x (ComplAP (Mod (Num y Year) Old))`

Construction

```
fun YearsOld : NP -> Numeral -> Cl
lin YearsOld x y = Pred x (ComplAP (Mod (Num y Year) Old))
```

Translating a construction

English-French data:

John is five years old	*John a cinq ans*
they are seventy years old	*ils ont soixante-dix ans*
I am fifteen years old	*j'ai quinze ans*
repeated pattern:	`Pred x (ComplV2 Avoir (Num y An))`

Construction

```
lin YearsOld x y = Pred x (ComplV2 Avoir (Num y An))
```

Translating with a construction

Add the new rules to the RGL

More ambiguity in parsing:

```
she is twenty years old ->
  Pred She  (ComplAP (Mod (Num 20 Year) Old))
  YearsOld She 20
```

However, the special construction gets a higher probability.

```
* elle est vieille de vingt ans
elle a vingt ans
```

The current GF systems

11 languages

web-based

mobile