Statistical Machine Translation Using Thot

Daniel Ortiz Martínez

March 2017

Table of Contents

1. Introduction

2. Modeling

3. Training

4. Search

5. Evaluation

6. Advanced Topics

7. Statistical Machine Translation with Thot

Introduction

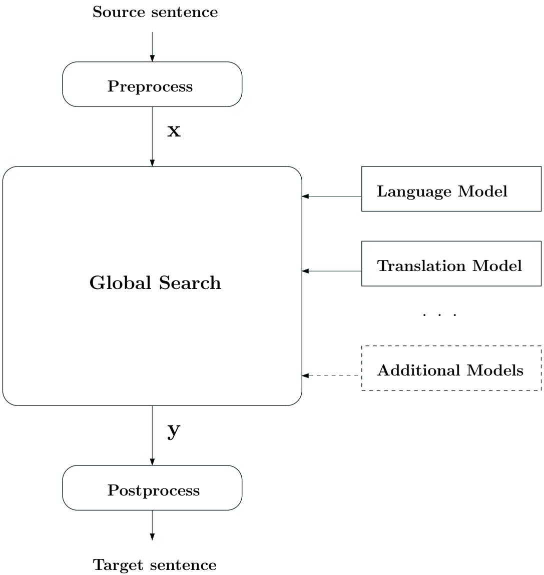
Statistical Machine Translation

- For a given source sentence $\boldsymbol{x}$, SMT finds the translation of highest probability in the target language, $\boldsymbol{y}$

$$\hat{\boldsymbol{y}} = \arg\max_{\boldsymbol{y}} \{Pr(\boldsymbol{y}|\boldsymbol{x})\} = \arg\max_{\boldsymbol{y}} \{Pr(\boldsymbol{y}) \cdot Pr(\boldsymbol{x}|\boldsymbol{y})\}$$

- SMT is based on statistical models
  - Measure the correctness of the translation
  - Trained on parallel corpora
- Given $\boldsymbol{x}$ and the models, $\boldsymbol{y}$ is obtained through a search process

Architecture of an SMT System



Pre/Post-processing

- Before translating, it is useful to *digest* the input text to make things easier to the translation system

- Common preprocessing tasks:
    - **Tokenization**: “Black ink cartridge.” → “Black ink cartridge .”
    - **Lowercasing**: “Black ink cartridge .” → “black ink cartridge .”

- Post-processing is necessary to obtain raw output text:
    - **Recasing**: “cartucho de tinta negro .” → “Cartucho de tinta negro .”
    - **Detokenization**: “Cartucho de tinta negro .” → “Cartucho de tinta negro.”

Modeling

Modeling

- **Language model**
  - Measures the fluency of the target sentence
  - Assigns better score to well formed target text
- **Translation model**
  - Measures the adequacy of the target sentence as a translation of the source sentence
  - Assigns better score to accurate and complete translations

n-gram Language Models

- *n*-gram models are a popular implementation of language models
- An *n*-gram is a vector of *n* consecutive words
- Assign scores to each word depending on the $n - 1$ preceding words
- They are estimated from target texts
- An *n*-gram model is basically a set of *n*-gram counts

Phrase Translation Models

- Phrase models are a common way to implement translation models

- Phrase-based translation follows a three step process:
  1. Divide the source sentence into segments
  2. Choose the target translations for each segment
  3. Reorder the target phrases to compose the final translation

- A phrase model is basically a dictionary of phrase pairs with scores

Phrase-based Translation Example

Step 1 (source segmentation):

$\mathbf{x}$: $\boxed{\text{material}}\boxed{\text{excelente}}\boxed{\text{para}}\boxed{\text{diversos usos}}$

Step 2 (phrase translation):

material → material

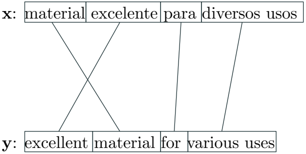
excelente → excellent

para → for

diversos usos → various uses

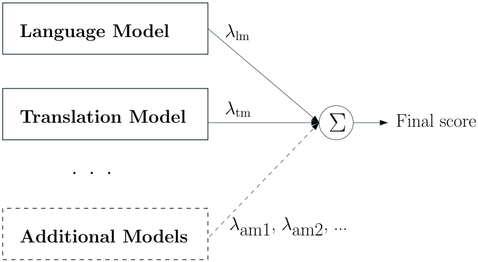
Step 3 (reordering):

$\mathbf{y}$: $\boxed{\text{excellent}}\boxed{\text{material}}\boxed{\text{for}}\boxed{\text{various uses}}$


x: material excelente para diversos usos
y: excellent material for various uses

Model Combination

- Common SMT systems rely on a combination of different models
- Language and translation models are the basis of the combination
- Additional models can be included
- Each model has a weight, λ, defining its importance



Training

Training

- SMT systems use training corpora to estimate model parameters
- Language models require monolingual data for the target language

Black ink cartridge for Canon
Pure grapefruit essential oil
Adidas FEF Spain shoe bag
...

- Phrase models require bilingual data

Cargador para portátil ACER Aspire	Laptop charger for ACER Aspire
Nuevo reloj TAG-HEUER Fórmula-1	New TAG-HEUER Formula-1 watch
Funda de almohada decorativa 40cm	16" decorative pillowcase
...	...

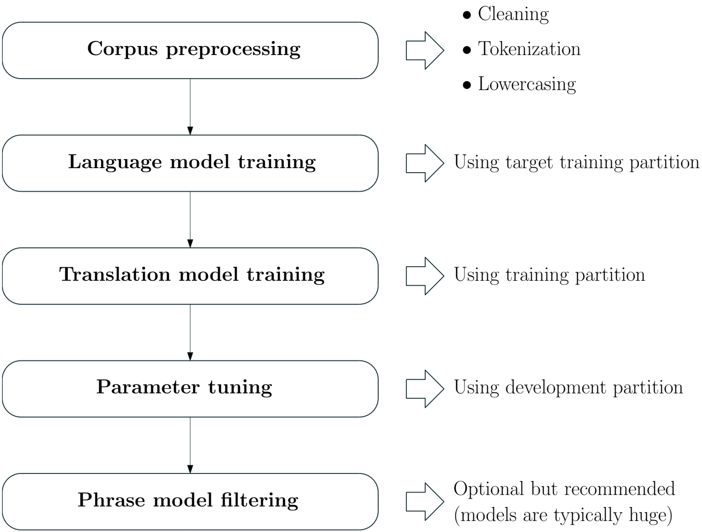
Training

- Translation quality is strongly affected by corpora availability
    - Corpus size is very important (the larger the better)
    - Corpus domain is also critical
- Models estimated from very large corpora are difficult to handle
    - Training is very time consuming
    - Huge amounts of memory are required to load them
    - Loading times can also be huge

Corpus Partition

- To carry out experiments, the training corpus is typically divided into three partitions:
  - **Training partition**: a large subset of the whole corpus which is used to train language and translation models
  - **Development partition**: a small portion (a few thousand sentences) useful to adjust the weights of the model combination
  - **Test partition**: a small subset (a few thousand sentences) of the corpus used to generate translations and evaluating the final quality

Training Pipeline



Corpus preprocessing

- Cleaning
- Tokenization
- Lowercasing

Language model training

Using target training partition

Translation model training

Using training partition

Parameter tuning

Using development partition

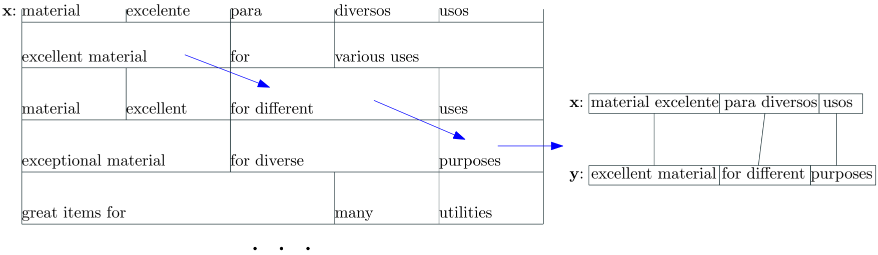
Phrase model filtering

Optional but recommended
(models are typically huge)

Search

Search

- After training the models, they can be used to generate translations
- Given a source sentence, **x**, it can be translated in many ways:



x: material excelente para diversos usos

excellent material | for | various uses

material | excellent | for different | uses

exceptional material | for diverse | purposes

great items for | many | utilities

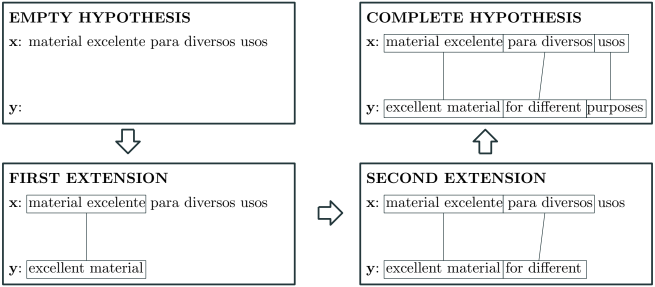
. . .

x: material excelente | para diversos | usos

y: excellent material | for different | purposes

Search

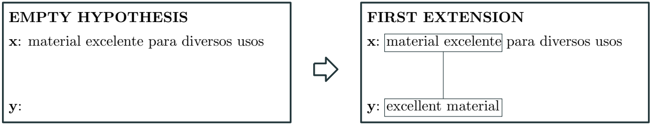
- The search space is explored by generating translation hypotheses
- Translation hypotheses are built in an incremental manner
- Partial hypotheses can be extended by adding words to them


EMPTY HYPOTHESIS
x: material excelente para diversos usos
y:
FIRST EXTENSION
x: material excelente para diversos usos
y: excellent material
SECOND EXTENSION
x: material excelente para diversos usos
y: excellent material for different
COMPLETE HYPOTHESIS
x: material excelente para diversos usos
y: excellent material for different purposes

Search

- SMT finds the translation of highest score according to the models

- The score of a partial hypothesis is revised after each extension



EMPTY HYPOTHESIS

x: material excelente para diversos usos

y:

FIRST EXTENSION

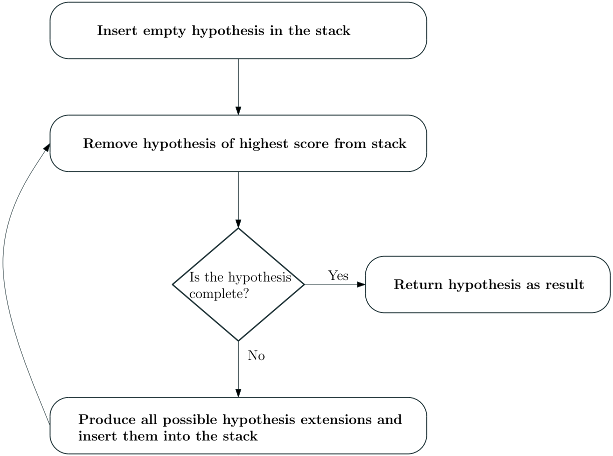
x: material excelente para diversos usos

y: excellent material

- Contributions to score for the previous example:
  - **Phrase model**: adds a score due to the translation of “material excelente” by “excellent material”
  - **Language model**: adds a score due to the addition of the words “excellent material”

Search Algorithm

- An iterative algorithm is used to reach the goal translation
- The algorithm uses a stack (priority queue) to organize the search


Insert empty hypothesis in the stack
Remove hypothesis of highest score from stack
Is the hypothesis complete?
Yes
Return hypothesis as result
No
Produce all possible hypothesis extensions and insert them into the stack

Evaluation

Evaluation

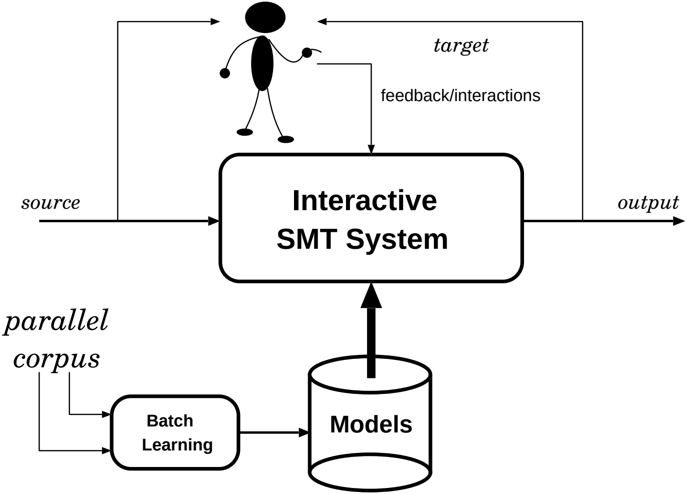
- Translation quality measures can be automatic or manual
- The test partition can be used to compute automatic measures using the target sentences as references
- Two common automatic measures:
  - **BLEU**: the BLEU (bilingual evaluation understudy) score is a quality measure based on *n*-gram precision for different values of *n* plus a brevity penalty
  - **WER**: the WER (word error rate) measure counts the number of substitutions, insertions and deletions required to convert the system translation into the reference sentence

Advanced Topics

Post-Editing and Interactive Machine Translation

- SMT allows us to translate a source text without human intervention
- Unfortunately, SMT results are not error-free
- SMT output can be supervised to obtain high-quality translations
- Two SMT applications allow users to collaborate with the system:
    - Post-editing (PE): sequential collaboration
    - Interactive Machine Translation (IMT): interactive collaboration

Interactive Machine Translation



Interactive Machine Translation Example

source($\boldsymbol{x}$): Para ver la lista de recursos

reference($\hat{\boldsymbol{y}}$): To view a listing of resources

interaction-0	$\boldsymbol{p}$							
	$\boldsymbol{s}$	*To*	*view*	*the*	*resources*		*list*	
interaction-1	$\boldsymbol{p}$	To	view					
	k			$\boxed{\textbf{a}}$				
	$\boldsymbol{s}$				*list*		*of*	*resources*
interaction-2	$\boldsymbol{p}$	To	view	a	list			
	k					$\boxed{\textbf{i}}$		
	$\boldsymbol{s}$					*ng*	*resources*	
interaction-3	$\boldsymbol{p}$	To	view	a	listing			
	k						$\boxed{\textbf{o}}$	
	$\boldsymbol{s}$						*f*	*resources*
acceptance	$\boldsymbol{p}$	To	view	a	listing		of	resources

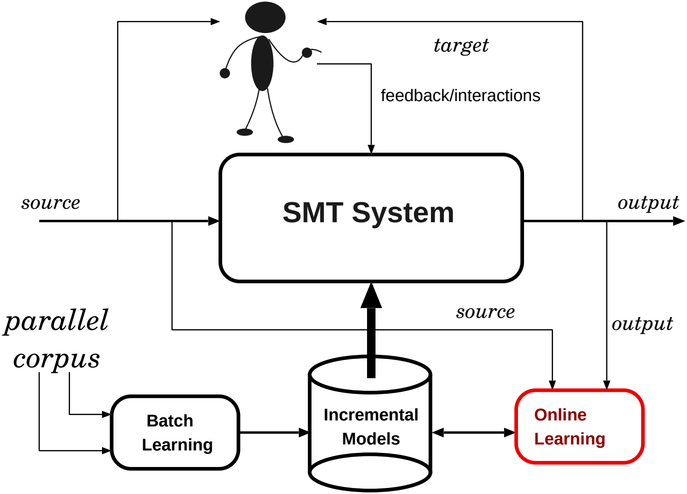
Online Learning

- Appropriate in those learning tasks in which learning must take place over time
- Examples are not available a priori but become available over time, usually one at a time
- Online learning is opposed to batch learning, where there is a finite set of examples that are available a priori

Main Features of Online Learning

- No re-processing of previous samples is required.
- The learner can, at any time, produce an answer to a query
- The quality of the answers improves over time

Online Learning for SMT



Statistical Machine Translation with Thot

Statistical Machine Translation with Thot

- Thot is a toolkit for phrase-based SMT
- Hosted on github: http://daormar.github.io/thot/
- Many features
  - Training, tuning and searching functionality
  - Can be executed in parallel on multiprocessors or clusters
  - Incorporates interactive machine translation and online learning
- Currently under development

Installation

- Obtain the package using git:

  ```
  git clone https://github.com/daormar/thot.git
  ```

- Change to the directory with the package's source code and type:

  ```
  ./reconf
  ./configure
  make
  make install
  ```

  NOTE: use `--prefix` option of `configure` to install the package in a custom directory

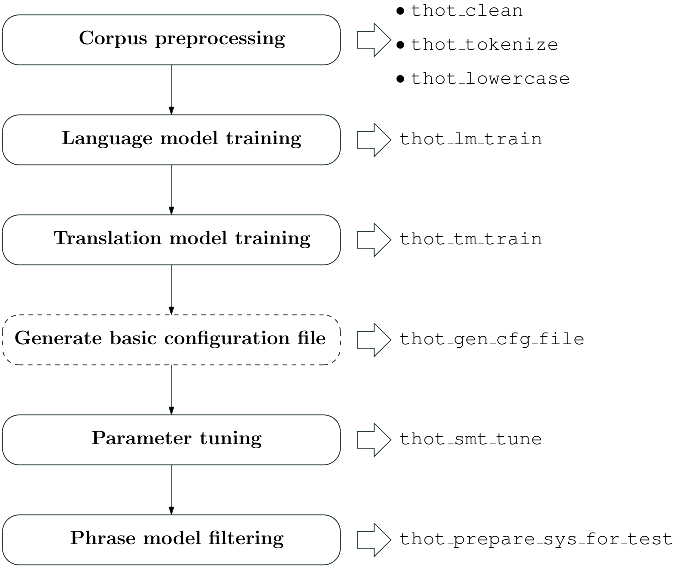
- Finally, after installation, the package can be checked by typing:

  ```
  make installcheck
  ```

File Naming Conventions

- To simplify the usage of some tools, a naming convention has been adopted for the files containing a corpus partition
- One example can be found in the Spanish to English toy corpus included with Thot:
    - `{sp}|{en}.train`: training partition
    - `{sp}|{en}.dev`: development partition
    - `{sp}|{en}.test`: test partition
- Additional conventions have been defined to name files containing tokenized (`tok` suffix) and lowercased (`lc` suffix) texts

SMT Pipeline and Thot Commands (I)



Corpus preprocessing

- thot_clean
- thot_tokenize
- thot_lowercase

Language model training ⇨ thot_lm_train

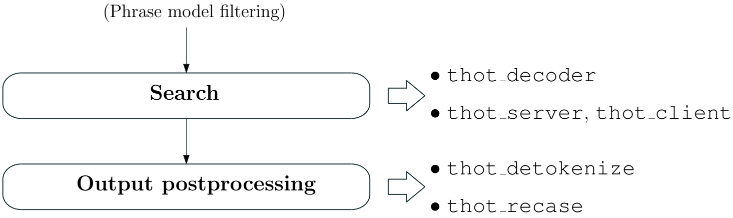
Translation model training ⇨ thot_tm_train

Generate basic configuration file ⇨ thot_gen_cfg_file

Parameter tuning ⇨ thot_smt_tune

Phrase model filtering ⇨ thot_prepare_sys_for_test

SMT Pipeline and Thot Commands (II)



(Phrase model filtering)

Search

- thot_decoder
- thot_server, thot_client

Output postprocessing

- thot_detokenize
- thot_recase

Thot Additional Commands

- `thot_auto_smt`: automates the whole SMT pipeline with one simple command
- `thot_calc_bleu`: computes the BLEU score
- `thot_calc_wer`: computes the WER measure
- ...

For additional information, check the Thot documentation

Questions?

daniel.ortiz.phd@gmail.com