Webinterpret

Big Data and Natural Language Processing

Daniel Ortiz Martínez

Talk at the School of Engineering of the University of Valencia
Valencia, 3 March 2017

Table of Contents

1. Introduction
2. Machine Translation
3. The Role of Data in NLP
4. Design and Implementation of NLP Applications
5. Conclusions

Introduction

Basic English and Natural Language Processing

- In 1930, C. K. Ogden proposed the so-called *basic English*
  - international auxiliary language, aid for teaching English
  - restricted grammar and a vocabulary of 1 000 words
- Still used today but harshly criticized
  - *choice of vocabulary*
  - grammatical constraints

Basic English and Natural Language Processing

- How to choose the vocabulary of basic English in the age of big data?
- Simplest proposal: rely on the concept of word frequency
- Procedure:
    1. calculate word frequencies extracted from large texts
    2. obtain text coverage provided by the N most frequent words
    3. select value of N according to a certain threshold

The Europarl Corpus

- The Europarl corpus is a set of parallel texts extracted from the proceedings of the European Union (Koehn 2005)
- Useful to build statistical machine translation systems
- English-Spanish set was downloaded and preprocessed
- Some figures about the English part:

Sentences:	1.9M
Words:	54M
Vocabulary:	120K

Obtaining Word Frequencies

1. Download corpus:

```
wget http://www.statmt.org/europarl/v7/es-en.tgz
tar -zxvf europarl-v7.es-en.tgz
```

2. Text preprocessing*:

```
thot_tokenize < europarl-v7.es-en.en > europarl-v7.es-en.en_tok
thot_lowercase < europarl-v7.es-en.en_tok > europarl-v7.es-en.en_tok_lc
```

3. Calculate frequencies:

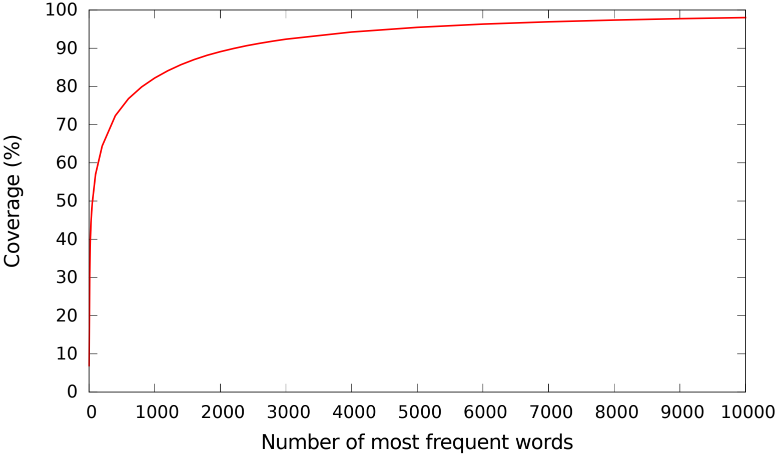
```
awk '{for(i=1;i<=NF;++i) print $i;}' europarl-v7.es-en.en_tok_lc | sort | uniq -c
```

*Thot toolkit preprocessing utilities are used in the example

The Ten Most Frequent English Words

Word	Frequency
the	3 769 336
of	1 785 835
to	1 657 883
and	1 416 334
in	1 168 692
that	902 118
a	833 678
is	831 418
for	576 254
we	568 536

Studying Corpus Coverage



- 1 000 words cover an 82% of the corpus
- 2 000 words cover an 89% of the corpus

A Simple Spell Checker

- Most popular method of detecting spelling errors is to look up every word in a dictionary
  - Words are verified but no corrections are proposed
  - Morphology can be handled as an additional language-dependent step
- Data-driven approach relies on large collections of texts
  - Texts can be used to obtain the dictionary
  - Lack of morphology analysis step is tackled with more data
  - Distance metrics can be used to suggest corrections

A Simple Spell Checker: Creating a Dictionary[†]

1. Download corpus:

```
wget http://www.statmt.org/europarl/v7/es-en.tgz
tar -zxvf europarl-v7.es-en.tgz
```

2. Text preprocessing:

```
thot_tokenize < europarl-v7.es-en.en > europarl-v7.es-en.en_tok
thot_lowercase < europarl-v7.es-en.en_tok > europarl-v7.es-en.en_tok_lc
```

3. Calculate frequencies:

```
awk '{for(i=1;i<=NF;++i) print $i;}' europarl-v7.es-en.en_tok_lc | sort | uniq -c
```

```
3769336 the
2595167 ,
1911462 .
1785835 of
1657883 to
1416334 and
1168692 in
   ...
```

[†] Same steps as in previous word frequency example

A Simple Spell Checker: Edit Distance

- Edit distance between two strings is the minimum number of edit operations required to transform one into the other
- Edit operations:
    - Insertion
    - Deletion
    - Substitution

N	*O*	–	–	–	*T*	*H*	*I*	*N*	*G*
≀	≀				|	|	|	|	|
E	*V*	*E*	*R*	*Y*	*T*	*H*	*I*	*N*	*G*
s	*s*	*i*	*i*	*i*					

E	*V*	*E*	*R*	*Y*	*T*	*H*	*I*	*N*	*G*
≀	≀	≀	≀		|	|	|	|	|
S	*O*	*M*	*E*	–	*T*	*H*	*I*	*N*	*G*
s	*s*	*s*	*s*	*d*					

A Simple Spell Checker: Verifying a Single Word

```
import editdistance

def obtain_edit_dist(lfile,word):
    eddist_list=[]
    for line in lfile:
        line=line.strip("\n")
        line_array=line.split()
        if(len(line_array)==2):
            list_freq=line_array[0]
            list_word=line_array[1]
            eddist=editdistance.eval(word,list_word)
            eddist_list.append((eddist,list_word,list_freq));
     return eddist_list

def check_word(lfile,word):
    eddist_list=obtain_edit_dist(lfile,word)
    sorted_eddist_list=sorted(eddist_list,key= lambda eddist_list: eddist_list[0])
    for elem in sorted_eddist_list[:10]:
        print elem[0],elem[1],elem[2]
```

A Simple Spell Checker: Some Examples

- Output for word *procesing*:

```
1 processing 1590
1 proceding 1
2 proposing 3459
2 promising 659
2 proceeding 495
2 protesting 226
2 preceding 203
2 procuring 25
2 professing 9
2 procedings 1
```

A Simple Spell Checker: Some Examples

- Output for word *behavior*:

```
0 behavior 1
1 behaviour 2549
2 behaving 199
2 behaviours 22
3 behave 486
3 behaved 163
3 heavier 93
3 behavioural 56
3 behaves 53
3 benazir 53
```

Natural Language Processing

- NLP is about interaction between computer and human languages
- Initial attempts required hand coding of large sets of rules
- Modern approach is based on (statistical) *machine learning*
  - uses algorithms that can learn and make predictions on data
  - a *model* is built from sample inputs (training set or corpus)
  - the model is later used to make predictions

Natural Language Processing: Example Tasks

- Spam detection
- Part of speech (POS) tagging
- Machine translation
- Sentiment analysis
- Text summarization
- Dialog

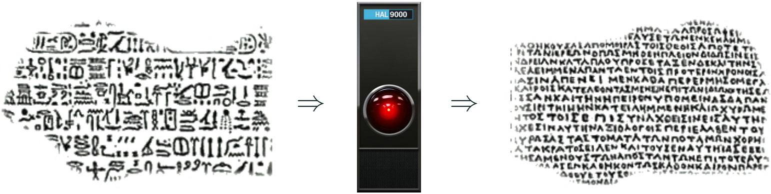
Natural Language Processing: Task Status[‡]

- **Mostly solved**:
  - Spam detection
  - Part of speech (POS) tagging
- **Good progress**:
  - Machine translation
  - Sentiment analysis
- **Still really hard**:
  - Text summarization
  - Dialog

[‡] See this video for additional information

Machine Translation

Machine Translation



- Translate from one language to another using a computer
- Origins of MT date back to 1950 and was the first NLP application
- Different approaches: rule-based versus corpus-based

Corpus-Based Machine Translation

- Corpus-based machine translation requires a set of translation examples from one language to another

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
...	...

- Types:
  - Example-based machine translation
  - *Statistical machine translation*
  - *Neural machine translation*

Statistical Machine Translation

- Seminal work conceived translation as decryption (Weaver 1949/1955)
- Translation formulated as a statistical process (Brown et al. 1993)
- Lack of linguistic intuition

Original Review of First Statistical Machine Translation Paper for Coling 1988

The validity of statistical (information theoretic) approach to MT has indeed been recognized, as the authors mention, by Weaver as early as 1949. And was universally recognized as mistaken by 1950. (cf. Hutchins, MT: Past, Present, Future, Ellis Horwood, 1986, pp 30 ff. and references therein).

The crude force of computers is not science. The paper is simply beyond the scope of COLING.

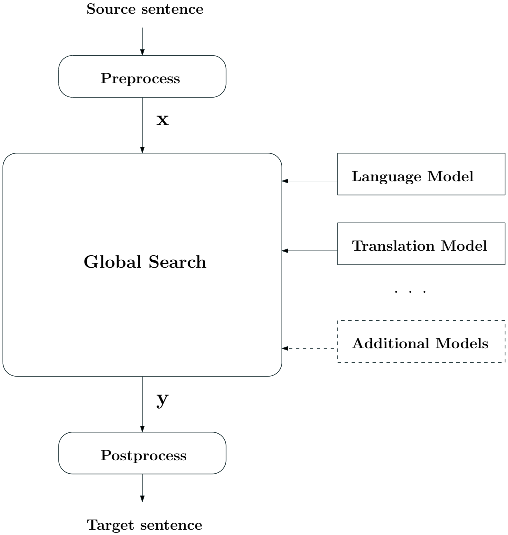
Statistical Machine Translation

- 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

SMT: Architecture



SMT: 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

SMT: *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
- An *n*-gram model is basically a set of *n*-gram counts

SMT: 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

SMT: Phrase-based Translation Example

Step 1 (source segmentation):

x: | material | excelente | para | diversos usos |

Step 2 (phrase translation):

material → material

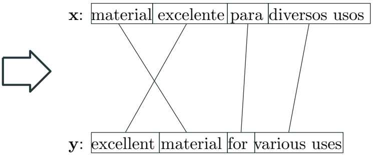
excelente → excellent

para → for

diversos usos → various uses

Step 3 (reordering):

y: | excellent | material | for | various uses |



x: | material | excelente | para | diversos usos |

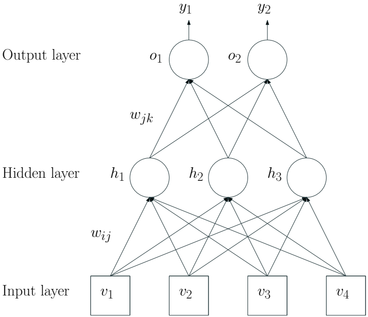
y: | excellent | material | for | various uses |

SMT: Pros and Cons

- **Pros**:
    - Easy to build
    - Scales well when new training data is added
- **Cons**:
    - Requires parallel texts
    - Computationally intensive

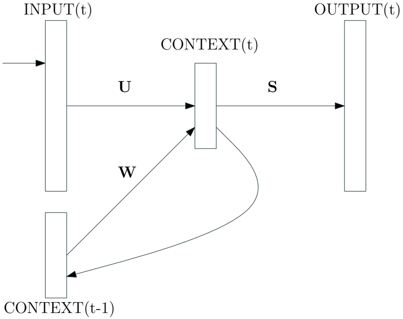
Neural Machine Translation

- Neural machine translation (NMT) is a machine translation approach based on neural networks trained by deep learning techniques
- Neural networks (NNs) are statistical learning algorithms that estimate functions from a set of inputs



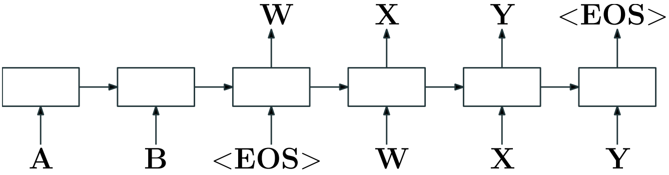
NMT: LMs based on Recurrent Neural Networks

- RNNs are particularly appealing to deal with context information
- At instant t, the input is given by the current word and the context in $t-1$



NMT: Architecture

- NMT systems use two concatenated RNN LMs (Sutskever et al. 2014)
- First RNN LM encodes source sentence into a vector
- Second RNN LM decodes this vector into the target sentence



NMT: Pros and Cons

- **Pros**:
  - Continuous representation of words
  - Better handling of contextual information
  - Simpler architecture than that of SMT systems
- **Cons**:
  - Computationally intensive (much more than SMT)
  - Alignment between source and target not explicitly modeled
    - Source words not translated
    - Source words translated multiple times

The Role of Data in NLP

Introduction

"*We don't have better algorithms than anyone else; we just have more data*"

Peter Norvig, Research Director at Google

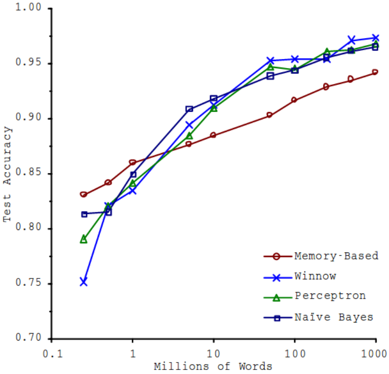
The Unreasonable Effectiveness of Data[§]

- Biggest successes in NLP applications such as machine translation or speech recognition are due to the availability of data
- Languages cannot be described with a small set of generative rules
- Instead, a big enough set of examples can be used
- Find the data that already exist
- "*More data beats clever algorithms*"

[§] See (Halevy et al. 2009)

The Unreasonable Effectiveness of Data: Example

- Learning curves for confusion set disambiguation (Banko and Brill 2001):



- Where to spend money: algorithm or corpus development?

The Unreasonable Effectiveness of Data: Criticism[¶]

- There are times when more data *does not* help
- Two learning scenarios:
  - **High variance**: model is too complicated for the amount of data we have (more data helps)
  - **High bias**: model is too simple to explain the data (more features help)
- Better data is not the same as more data
- Data without a sound approach becomes noise

[¶] for more details follow this link

Design and Implementation of NLP Applications

Main Elements of an NLP Application

- Data
- Algorithms
- *Evaluation*

Data

- Which kind of data do I need?
- Do I have enough data?
- How can I obtain new data?
- Do I need all the data I have?

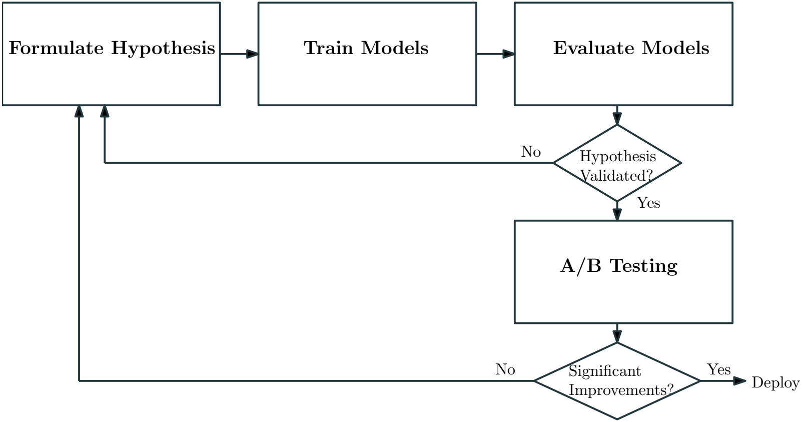
Algorithms

- How much data are available?
- Are the data labeled?
- Do I need to predict categories or quantities?
- How complex is the task?
- Is efficiency a critical aspect?

Evaluation

- Evaluation is as critical as data or algorithms
- An NLP system is always an *unfinished project*
- Evaluation provides the path to follow so as to improve performance
- Evaluation may have both a research or a bussiness perspective

Evaluation


Formulate Hypothesis
Train Models
Evaluate Models
No
Hypothesis Validated?
Yes
A/B Testing
No
Significant Improvements?
Yes
Deploy

Scalability

- A scalable NLP system should deal with an increasing workload
    - Concurrent user queries
    - Large training data sets
    - Large models
- Common problems:
    - Model training too slow
    - Models don't fit into memory
    - Insufficient system throughput
- Some hints:
    - Appropriate data structures (trie, suffix array)
    - Appropriate algorithms (dynamic programming, map-reduce)
    - Accessing models from disk

Implementing NLP Applications

- Reminder: an NLP application is always an unfinished project
  - Performance may need to be improved
  - Requirements may change
- Some important aspects to pay attention to when coding:
  - Modularity
  - Extensibility
  - Testability
  - Readability

Implementing NLP Applications

- Reminder: an NLP application is always an unfinished project
  - Performance may need to be improved
  - Requirements may change
- Some important aspects to pay attention to when coding:
  - Modularity
  - Extensibility
  - Testability
  - Readability
- Wait a minute! What about efficiency?

Implementing NLP Applications

- Reminder: an NLP application is always an unfinished project
    - Performance may need to be improved
    - Requirements may change
- Some important aspects to pay attention to when coding:
    - Modularity
    - Extensibility
    - Testability
    - Readability
- Wait a minute! What about efficiency?

> "*Premature optimization is the root of all evil in programming*"
>
> Donald Knuth

Conclusions

Conclusions

- Data availability explains great advances in NLP applications
- “More data beats clever algorithms”
- Invest money in corpus development or in algorithm development?
- Better data is not the same as more data
- Data are useless without sound algorithms

References i

Banko, Michele and Eric Brill (2001). “Scaling to Very Very Large Corpora for Natural Language Disambiguation”. In: *Proceedings of 39th Annual Meeting of the Association for Computational Linguistics*. Toulouse, France: Association for Computational Linguistics, pp. 26–33.

Brown, Peter F., Stephen A. Della-Pietra, Vincent J. Della-Pietra, and Robert L. Mercer (1993). “The Mathematics of Statistical Machine Translation”. In: *Computational Linguistics* 19.2, pp. 263–313.

Halevy, Alon, Peter Norvig, and Fernando Pereira (2009). “The Unreasonable Effectiveness of Data”. In: *IEEE Intelligent Systems* 24.2, pp. 8–12. ISSN: 1541-1672.

Koehn, Philipp (2005). “Europarl: A Parallel Corpus for Statistical Machine Translation”. In: *Conference Proceedings: the tenth Machine Translation Summit*. AAMT. Phuket, Thailand, pp. 79–86.

References ii

Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le (2014). "Sequence to Sequence Learning with Neural Networks". In: *Proceedings of the 27th International Conference on Neural Information Processing Systems*. NIPS'14. Montreal, Canada: MIT Press, pp. 3104–3112.

Weaver, Warren (1949/1955). "Translation". In: *Machine Translation of Languages*. Ed. by William N. Locke and A. Donald Boothe. Reprinted from a memorandum written by Weaver in 1949. Cambridge, MA: MIT Press, pp. 15–23.

Questions?