Rule Based, Statistical and Neural Methods for NLP

Carlos Escolano
carlos.escolano@tsc.upc.edu

PhD. Candidate
Universitat Politecnica de Catalunya
Technical University of Catalonia

[course site]

Outline

Problem Requirements

Rule Based Methods

Statistical Methods

Neural Methods

Hybrid methods

Wrap Up

Example

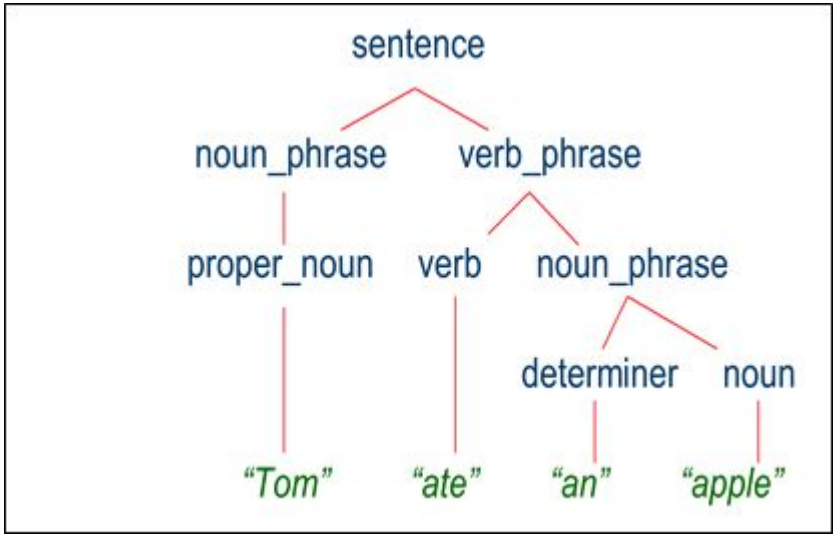
Problem Requirements

Problem Requirements.

Before starting an NLP project we should ask ourselves the following questions:

- What's the task to solve?
- Do we have expertise in the domain?
- Do we have data available? How much?
- What hardware can we use?

Problem Requirements.



Parsing

Find the underlying structure of a sentence and the words that form it.

Problem Requirements.

私の名前はCARLOSです。-> My name is Carlos.

My name is Carlos –> Mi nombre es Carlos.

私の名前はCARLOSです。-> Mi nombre es Carlos.

Machine Translation

- How many resources are available on that language?
- From resources how many are from the domain we are working on?

Text processing

Text Processing

- How to separate the words in the sentence?
- How to manage the different forms of a word?
- How to match all instances of a word as the same token?

The kid keeps a secret.

The kids are keeping secrets.

Is John keeping a secret?

Text Processing

- **Tokenization:** Split a sentence into different tokens, e.g words, subwords, characters.

[The, kid, keeps, a, secret, .]

- **Punctuation:** Punctuation marks depending on the needs of the task may need:
  - **Remove stop words:** For classification tasks they can be removed to reduce sequence lengths and vocabulary
  - **Normalize punctuation:** For generation task ensure that signs are correctily set in a consistent way

Text Processing

- **Casing:** Another form of ambiguity is its casing, two options depending of the task:
  - **Lower case:** Use lower case for all words in the sentence.
  - **True Casing:** Learn a model of words that appear with an initial Upper case in the corpus. Using this model, after computation the original casing can be recovered

Text Processing

- Manage different forms of the same word:
  - **Stemming:** Truncating the word to the common part between all the different words.
  - **Lemmatization:** Replacing all forms of the word by its basic shared form (lemma)

Having, Had, Has

Stemming: Ha
Lemmatization: Have

Rule Based Methods

Rule based methods

- Crafted by human experts
- Modelled following the process a human would take to solve the task
- Given they are human defined they are fully interpretable
- Grammar rules can be adapted to new domains easily
- No training corpus is required

Rule based methods

- Requires an expert to encode the rules in the system
- Longer deployment time as the systems has to me manually created
- The system can potentially grow to a point when it’s not easy to maintain

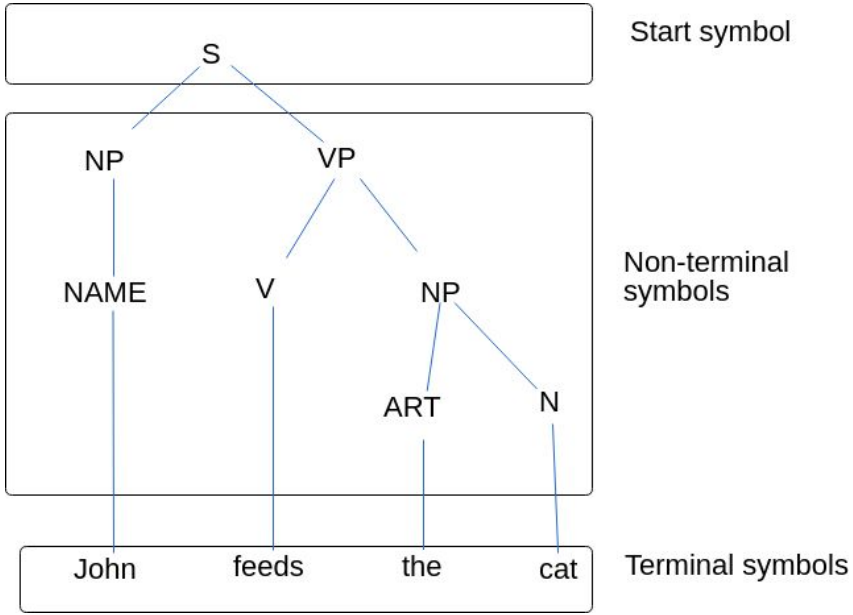
Context free grammars(CFG)

Definition: A CFG is a 5-tuple (P, A,N, T, S) where:

- P is a set of context-free productions, i.e. objects of the form X -> beta, where X is a member of N, and beta is a string over the alphabet A
- A is an alphabet of grammar symbols;
- N is a set of **non-terminal** symbols;
- T is a set of **terminal** symbols (and N U T = A);
- S is a distinguished non-terminal called the start symbol (think of it as "sentence" in NLP applications).

CFG: Example

Start symbol	s
Non-terminal symbols	NP VP NAME ART N
Terminal symbols	John feeds the cat
Non-terminal productions	1. S -> NP VP 2. VP-> V NP 3. NP-> NAME 4. NP-> ART N
Terminal productions	5. NAME -> John 6. V -> feeds 7. ART -> the 8. N -> cat



Should I use it for my problem?

For:

- No data required
- Can model specific domain necessities
- Interpretability

Against:

- Ambiguity is complex to model with rules
- Requires a lot of work to scale up
- Time costly to build
- Brittleness with non prepared uses of words.
- Requires human knowledge

Statistical methods

Statistical methods

Statistical methods model the probability distribution of a problem given a corpus of text, either by selecting features of the individual tokens or groups of continuous ones (n-grams).

Probabilistic Context Free Grammars(PCFG)

Which of these sentences is more common?

- It’s a chocolate cake
- It’s an onion cake

How to decide which one is the most plausible one?

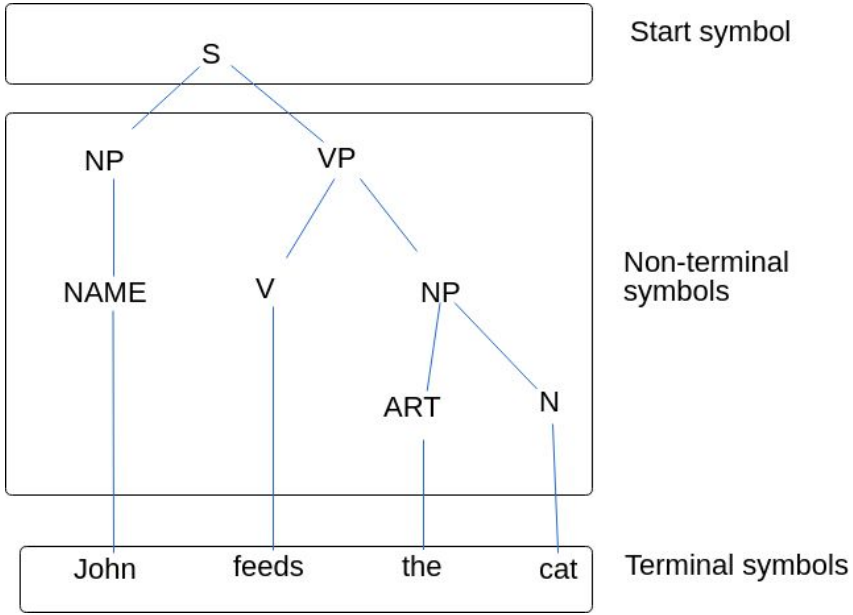
Probabilistic Context Free Grammars(PCFG)

Definition: A CFG is a 6-tuple (P, A, N, T, S, PR) where:

- M is a set of context-free productions, i.e. objects of the form X -> beta, where X is a member of N, and beta is a string over the alphabet A
- A is an alphabet of grammar symbols;
- N is a set of **non-terminal** symbols;
- T is a set of **terminal** symbols (and N U T = A);
- S is a distinguished non-terminal called the start symbol (think of it as "sentence" in NLP applications).
- PR probability associated to each production. Learned from data

PCFG: Example

Start symbol	s
Non-terminal symbols	NP VP NAME ART N
Terminal symbols	John feeds the cat
Non-terminal productions	1. S -> NP VP (1.0) 2. VP-> V NP (1.0) 3. NP-> NAME (0.4) 4. NP-> ART N (0.6)
Terminal productions	5. NAME -> John (1.0) 6. V -> feeds (1.0) 7. ART -> the (1.0) 8. N -> cat (1.0)



Hidden Markov Models (HMM)

Hidden Markov models (HMMs) are a way of relating a sequence of observations to a sequence of hidden classes or hidden states that explain the observations.

HMM:Markov's Assumption

A probabilistic model holds this property if the future state of the model depends exclusively on the present state of the model.

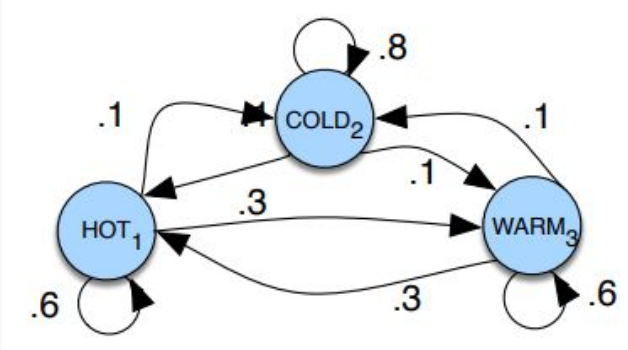
$$P(q_i = a|q_{i-1}, ..., q_1) = P(q_i = a|q_{i-1})$$

Example:

- Predict the next word of a sentence given the previous n words in the sentence

HMM: Markov Chains

- A Markov Chain models the relationship between some states by strongly assume that the next step is only dependent of the current one (Markov assumption)
- Each possible state defined as a node and each transition as an edge with an associated probability.



Initial probabilities

$$\pi = [0.2, 0.7, 0.1]$$

Source: https://web.stanford.edu/~jurafsky/slp3/A.pdf

HMM Definition

- Q: A set of states
- A: Transition matrix, defining the probability from one state to the others
- O: A set of observations
- B: Set of observations likelihoods.

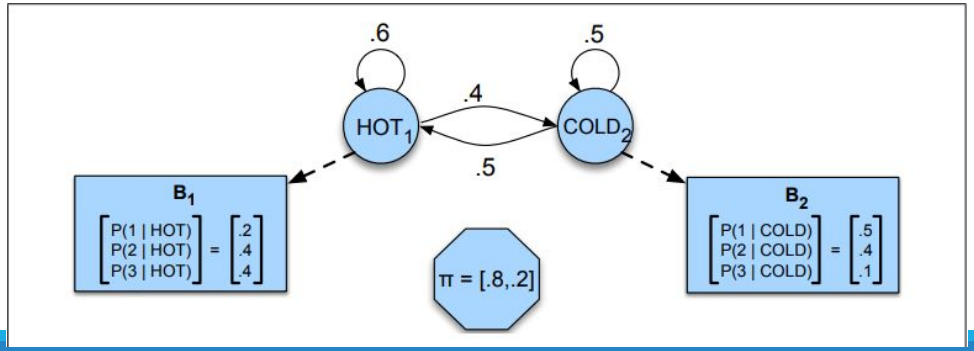
Tasks:

- Likelihood: Given a sequence how likely it is to occur
- Decoding: Given an observation generate a new sequence
- Learning: Given and observation O and the sets A and B learn the set of states of the HMM

HMM: Basic Example

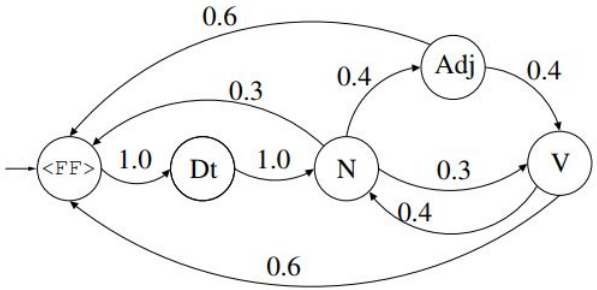
How to predict the temperature of a day given how many ice-creams has a store sold that day?

- States: Hot or Cold Day
- Observations: 1, ,2 or 3 ice creams sold



Source: https://web.stanford.edu/~jurafsky/slp3/A.pdf

HMM: Part of Speech tagging



Emission probabilities	.	el	la	gato	niña	come	corre	pescado	fresco	pequeña	grande
<FF>	1.0										
Dt		0.6	0.4								
N				0.6	0.1			0.3			
V						0.7	0.3				
Adj									0.3	0.3	0.4

Source: http://adimen.si.ehu.es/~rigau/research/Doctorat/LSKBs/AMCPLluisP.pdf

Moses

- Statistical Machine translation toolkit.
- Phrase-based.
- Good performance on low resources languages.

Source:
http://www.statmt.org/moses/

Feature extraction

- How to represent the words in our text to apply machine learning?
- Do we have additional information of the domain?
  - Dictionaries.
  - Ontologies.
- Do all words give the same amount of information?
  - Lemmatization.
  - Stemming.
  - Stop Words removal.

Bag of words

- Text is represented as a vector of size the number of words in the vocabulary. Position i represents the appearance of the ith word on the vocabulary.
- It produces sparse vectors as most of the words do not appear at the seem time in a piece of text.

Source: https://machinelearnings.co/text-classification-using-neural-networks-f5cd7b8765c6

Term Frequency-Inverse Document Frequency (TF-IDF)

- Measures the number of times that words appear in a given document (that's "term frequency")
- Words common to all document might not be relevant
- Words frequent in just a few documents may define the topic of such documents.

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

tf_{ij} = number of occurrences of i in j
df_i = number of documents containing i
N = total number of documents

Source:

Should I use it for my problem?

For:

- Good performance with small corpus
- Few hardware requirements.
- Strong baseline before using more costly methods

Against:

- Strong assumptions over the data.
- Domain dependant.
- Data quality dependant.

Neural methods

Distributed Representations

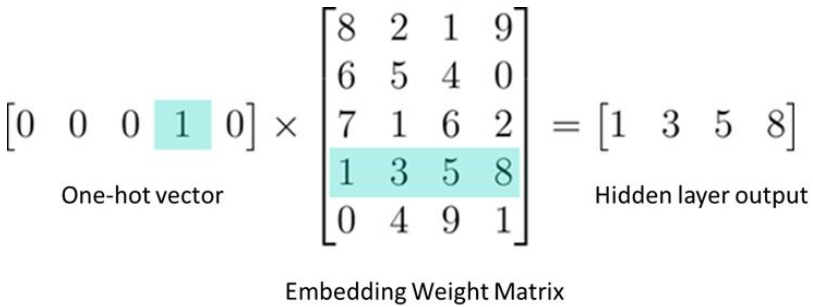
Bag of words representations present some problems:

- Discrete representation.
- Sparse vectors.
- No positional information

How to transform these representations into continuous vectors?

Embeddings

- For each token a vector representation is assigned
- Different granularities: Words, Subwords, Characters



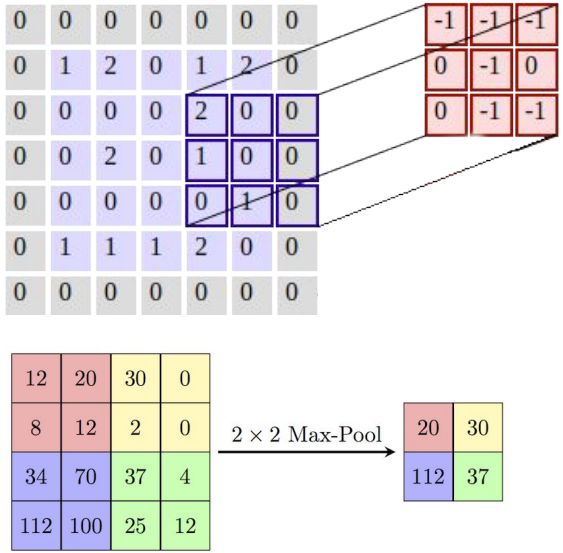
Embeddings

- **Trained with the network:** The embedding table is set as the first layer of the model. This layer is initialized as a matrix of size num_tokens x embedding_size with random weights that will be optimized during training.

- **Pretrained embeddings:** The embeddings are trained in another model in an unsupervised way. Examples: Word2Vec, Glove

Convolutional Neural Network (CNN)

- **CNN:** learns a set of filers that are applied over the input window by window to find local relations.
- **Pooling:** Reducing dimensionality by applying an assumption over windows of the data, e.g maximum, average

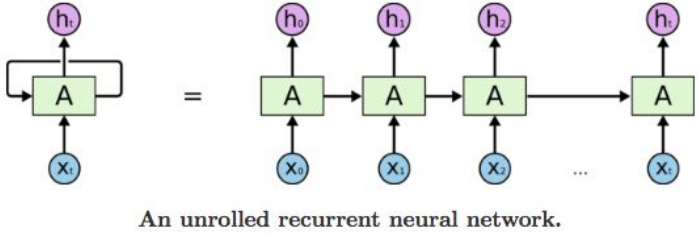


Source: http://cs231n.github.io/convolutional-networks/

Recurrent Neural Networks (RNN)

RNNS keep a context of the previous steps on the sequence that conditions the output of each steps.

This feature is useful for predicting new steps of the sequence or if only the last step is passed to the next layer to compute a sequence representation.

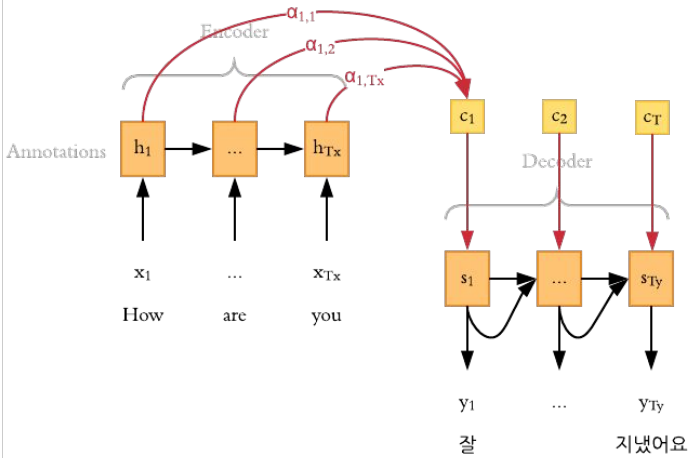


An unrolled recurrent neural network.

Source: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

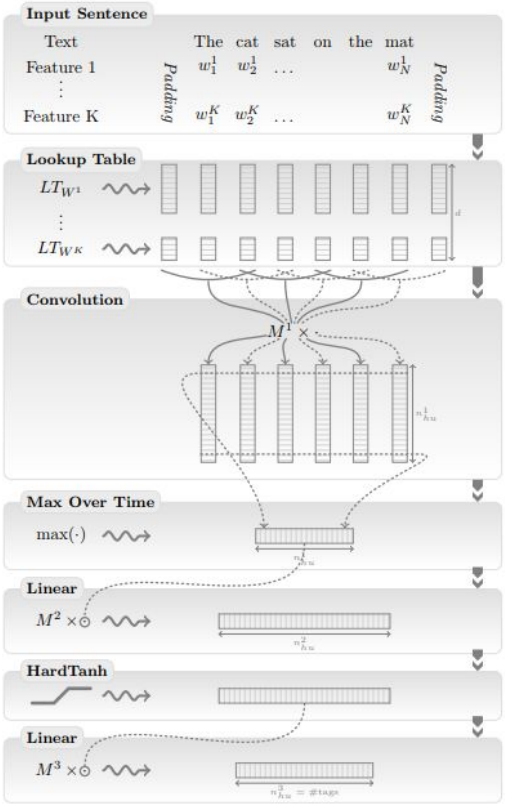
Attention mechanisms

Instead of retain information about the whole sequence focus on certain parts of the input for each token.



Source:
https://medium.com/@joealato/attention-in-nlp-734c6fa9d983

Example



Source: https://dl.acm.org/citation.cfm?id=2078186

Should I use it for my problem?

For:

- Current state of the art
- Good results without feature engineering

Against:

- Requires a lot of data
- Requires a lot of hardware
- No interpretability of intermediate representations

Hybrid methods

Monoses

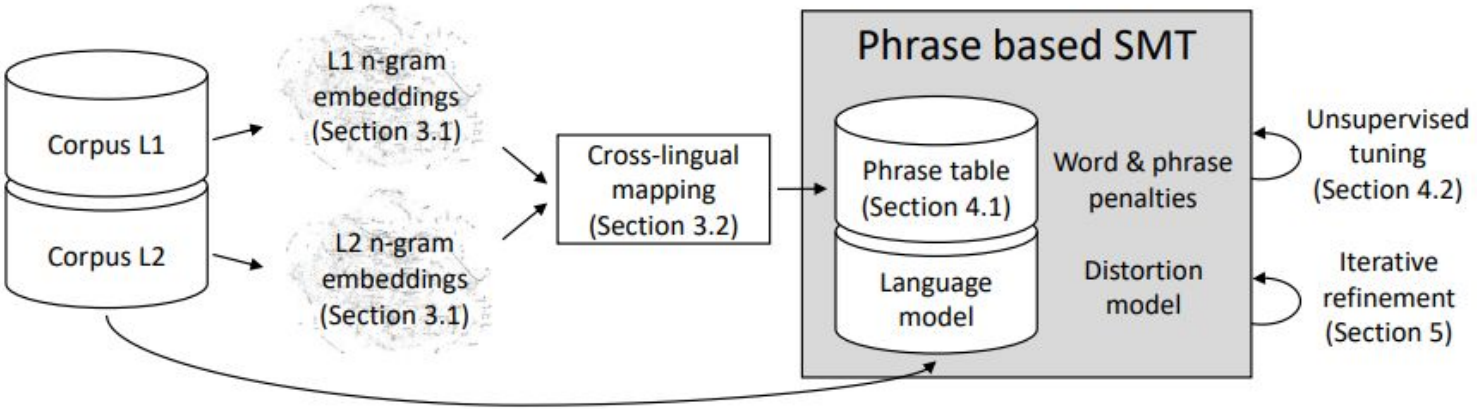


Figure 1: Architecture of our system, with references to sections.

Source: https://arxiv.org/pdf/1809.01272.pdf

Wrap Up

Wrap Up

- **Rule based methods:** Based on the knowledge of human experts. Long time to create. No data required. Adaptable to domains but can be difficult to manage when it grows.
- **Statistical Methods:** Models a probability distribution given a dataset. Strong assumptions over the data.
- **Neural Methods:** A lot of hardware and data required. Current state of the art in most tasks.
- **Conclusion:**The best method to use depends on the problem to solve.

Thanks ! Q&A ?

DEMO

You can download the demo and slides from: https://bit.ly/2XMxJjD