Natural Language Processing: Interpretation, Reasoning and **Machine Learning**

Roberto Basili  
(Università di Roma, Tor Vergata)  
dblp: [http://dblp.uni-trier.de/pers/hd/b/Basili:Roberto.html](http://dblp.uni-trier.de/pers/hd/b/Basili:Roberto.html)  
Google scholar: [https://scholar.google.com/citations?user=U1A22fYAAAAJ&hl=it&oi=sra](https://scholar.google.com/citations?user=U1A22fYAAAAJ&hl=it&oi=sra)

Politecnico di Milano, 4-5 Maggio 2017
Overview

• Artificial Intelligence, Natural Language & Speech
  • Information, Representation, (re)current challenges, success(and unsuccess)ful stories

• Natural Language Processing: linguistic background

• break

• Natural Language Processing: Tasks, Models and Methods

• The Role of Machine Learning Technologies
  • Lexicon Acquisition : Automatic Development of Dictionaries, Semantic Lexicons and Ontologies
  • Statistical Language Processing: Semantic Role Labeling

• break

• Natural Language Processing : Results & Applications
  • Semantic Document Management
  • Web-based Opinion Mining Systems, Market Watch & Brand Reputation Management
  • Human Robotic Voice Interaction
ML in NLP ... a prologue

• The syntax-semantic mapping

- Different semantic theories (e.g. PropBank vs. FrameNet)
• *Police arrested the man for shoplifting*
## A tabular vision

<table>
<thead>
<tr>
<th>Word</th>
<th>Predicate</th>
<th>Semantic Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Police</td>
<td>-</td>
<td>Authority</td>
</tr>
<tr>
<td>arrested</td>
<td>Target</td>
<td>Arrest</td>
</tr>
<tr>
<td>the</td>
<td>-</td>
<td>SUSPECT</td>
</tr>
<tr>
<td>man</td>
<td>-</td>
<td>SUSPECT</td>
</tr>
<tr>
<td>for</td>
<td>-</td>
<td>OFFENSE</td>
</tr>
<tr>
<td>Shoplifting</td>
<td>-</td>
<td>OFFENSE</td>
</tr>
</tbody>
</table>
Using Framenet/PropBank

SRL Pipeline

Syntactic Parse

Prune Constituents

Arguments

Argument Identification

Argument Classification

Structural Inference

Semantic Roles

Candidates

NP₁ Agent/Patient
V Predicate
PP Location/Patient

NP₁ Yes
VP No
V given
PP Yes
NP₂ No

V Walked
PP in
NP₂ the park
NLP: linguistic levels

- speech
- phonetics
- phonology

"shallower"

- morphology
- syntax

"deeper"

- semantics
- pragmatics
- discourse
Language as a system of rules

... comincia qui la mia disperazione di scrittore. Ogni **linguaggio** è un alfabeto di simboli il cui uso presuppone un passato che gli interlocutori condividono; come trasmettere agli altri l’infinito Aleph che la mia timorosa memoria a stento abbraccia?

- ... Meaning is acquired and recognized within the daily practices related to its usage
  - *The meaning of a word is to be defined by the rules for its use, not by the feeling that attaches to the words*
    

- Recognizing one meaning consists in the ability of mapping a linguistic expression to an experience (praxis) through mechanisms such as analogy or approximating equivalence functions or through the minimization of the risks of being wrong/inappropriate/obscure

- The interpretation process can be obtained through the induction of one (or more) decision function(s) from experience
The inductive process
The inductive process

1. Input documents
2. Annotation Citations
3. Testi Ann.
4. Modello Analisi
5. SVM Learning
6. Parole
7. Sintagmi
8. Alberi
9. FattiNoti

Flowchart showing the process from input documents through annotation, testing, modeling, and learning steps.
The inductive process

- Annotazione Fenomeni
- Testi
- Annotazioni
- Riconoscimento
- Modello
- SVM Learning
- Kernel_{Parole}
- Kernel_{Sintagmi}
- Kernel_{Tree}
- Kernel_{FattiNoti}
The inductive process

- Annotazione Fenomeni
- Testi
- Citazioni
- Riconoscimento
- Modello
- SVM Learning
- Kernel_{Parole}
- Kernel_{Sintagmi}
- Kernel_{Tree}
- Kernel_{FattiNoti}
The inductive process

Annotazione Fenomeni → Testi → Kernel_{Parole} → Kernel_{Sintagmi} → Kernel_{Tree} → Kernel_{FattiNoti} → SVM Learning

Riconoscimento

Citazioni

Modello
The inductive process
The inductive process

- Annotazione Fenomeni
- Testi
- Citazioni
- Riconoscimento
- Modello
- SVM Learning
- Kernel_{Parole}
- Kernel_{Sintagmi}
- Kernel_{Tree}
- Kernel_{FattiNoti}
IBM Watson: between Intelligence and Data

- IBM’s Watson

Jeopardy!
Hedgehogs are covered with quills or spines, which are hollow hairs made stiff by this protein.
Semantic Inference in Watson QA

In May 1498 Portugal celebrated the 400th anniversary of this explorer’s arrival in India.

On the 27th of May 1498, Vasco da Gama landed in Kappad Beach.

Several Inference Algorithms:
- Temporal Inference
- Statistical Paraphrasing
- GeoSpatial Inference
  - Date Math
  - Paraphrases
  - Geo-KB

Key terms:
- celebrated
- Portugal
- May 1898
- 400th anniversary
- arrival in
- India
- explorer
- landed in
- 27th May 1498
- Kappad Beach
- Vasco da Gama
... Intelligence in Watson
Watson: a DeepQA architecture
Ready for Jeopardy!
Strongly positive aspects

- Adaptivity of the overall workflow
- Significant exploitation of available data
- Huge volumes of knowledge involved

Criticalities

- The encyclopedic knowledge needed for Jeopardy is quite different in nature from the domain expertise required in many applications
- Wason is based on Factoid Questions strongly rooted on objective facts, that are explicit and non subjective
- Formalizing the input knowledge, as it is done a priori for Watson, is very difficult to achieve in cost-effective manner: sometimes such knowledge is even absent in an enterprise
- For many natural languages the amount of information and resources is not available, so that a purely data-driven approach is not applicable
Machine Learning: the weapons

- Rule and Pattern learning from Data
  - Frequent Pattern Mining (Basket analysis)

- Probabilistic Extensions of Grammars
  - Probabilistic CFGs
  - Stochastic Grammars

- Discriminative learning in neural networks

- SVM: perceptrons
  - Kernel functions in implicit semantic spaces

- Bayesian Models & Graphical Models
Figure 13.2 Two parse trees for an ambiguous sentence. The transitive parse (a) corresponds to the meaning “The book the dinner flight” and the intransitive parse (b) to “The book dinner the flight”.
\[ p(A, B, C, D, E) = p(A)p(B)p(C|A, B)p(D|B, C)p(E|C, D) \]
Hidden Markov Models (HMM)

- States = Categories/Classes
- Observations
- Emissions
- Transitions

Applications:
- Speech Recognition
- Sequence Labeling (e.g. POS tagging)

\[ p(X_1,\ldots,T, Y_1,\ldots,Y_T) = p(X_1)p(Y_1|X_1) \prod_{t=2}^{T} [p(X_t|X_{t-1})p(Y_t|X_t)] \]
The task of POS tagging

Given a sequence of morphemes $w_1, ..., w_n$ with ambiguous syntactic descriptions (i.e. part-of-speech tags) $t_j$, compute the sequence of $n$ POS tags $t_{j_1}, ..., t_{j_n}$ that characterize correspondingly all the words $w_i$.

Examples:

- Secretariat is expected to race tomorrow
- $\Rightarrow$ NNP VBZ VBN TO VB NR
- $\Rightarrow$ NNP VBZ VBN TO NN NR
The task of POS tagging

An example

Emission probabilities

<table>
<thead>
<tr>
<th></th>
<th>the</th>
<th>this</th>
<th>cat</th>
<th>kid</th>
<th>eats</th>
<th>runs</th>
<th>fish</th>
<th>fresh</th>
<th>little</th>
<th>big</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;FF&gt;</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dt</td>
<td>0.6</td>
<td>0.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td>0.6</td>
<td>0.1</td>
<td>0.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>V</td>
<td>0.7</td>
<td>0.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adj</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.3</td>
<td>0.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Given a sequence of morphemes $w_1, \ldots, w_n$ with ambiguous syntactic descriptions (i.e. part-of-speech tags), derive the sequence of $n$ POS tags $t_1, \ldots, t_n$ that maximizes the following probability:

$$P(w_1, \ldots, w_n, t_1, \ldots, t_n)$$

that is

$$(t_1, \ldots, t_n) = \operatorname{argmax}_{pos_1, \ldots, pos_n} P(w_1, \ldots, w_n, pos_1, \ldots, pos_n)$$

Note that this is equivalent to the following:

$$(t_1, \ldots, t_n) = \operatorname{argmax}_{pos_1, \ldots, pos_n} P(pos_1, \ldots, pos_n|w_1, \ldots, w_n)$$

as:

$$\frac{P(w_1, \ldots, w_n, pos_1, \ldots, pos_n)}{P(w_1, \ldots, w_n)} = P(pos_1, \ldots, pos_n|w_1, \ldots, w_n)$$

and $P(w_1, \ldots, w_n)$ is the same for all the sequencies $(pos_1, \ldots, pos_n)$. 
HMM and POS tagging

How to map a POS tagging problem into a HMM

The above problem

$$(t_1, \ldots, t_n) = \arg\max_{p_{os_1}, \ldots, p_{os_n}} P(p_{os_1}, \ldots, p_{os_n} | w_1, \ldots, w_n)$$

can be also written (Bayes law) as:

$$(t_1, \ldots, t_n) = \arg\max_{p_{os_1}, \ldots, p_{os_n}} P(w_1, \ldots, w_n | p_{os_1}, \ldots, p_{os_n})P(p_{os_1}, \ldots, p_{os_n})$$
The HMM Model of POS tagging:

- **HMM States are mapped into POS tags** \( (t_i) \), so that
  \[
P(t_1, \ldots, t_n) = P(t_1)P(t_2|t_1)\ldots P(t_n|t_{n-1})
  \]

- **HMM Output symbols are words**, so that
  \[
P(w_1, \ldots, w_n|t_1, \ldots, t_n) = \prod_{i=1}^{n} P(w_i|t_i)
  \]

- Transitions represent moves from one word to another

Note that *the Markov assumption is used*

- to model probability of a tag in position \( i \) (i.e. \( t_i \)) only by means of the preceding part-of-speech (i.e. \( t_{i-1} \))
- to model probabilities of words (i.e. \( w_i \)) based only on the tag \( (t_i) \) appearing in that position \( (i) \).
The final equation is thus:

$$(t_1, \ldots, t_n) = \arg\max_{t_1, \ldots, t_n} P(t_1, \ldots, t_n | w_1, \ldots, w_n) = \arg\max_{t_1, \ldots, t_n} \prod_{i=1}^n P(w_i | t_i) P(t_i | t_{i-1})$$
Fundamental Questions for HMM in POS tagging

1. Given a model what is the probability of an output sequence, \( O \): 
   *Computing Likelihood.*

2. Given a model and an observable output sequence \( O \) (i.e. words), how to determine the sequence of states \( (t_1, ..., t_n) \) such that it is the best explanation of the observation \( O \):  
   *Decoding Problem*

3. Given a sample of the output sequences and a space of possible models how to find out the best model, that is the model that best explains the data:  
   *how to estimate parameters?*
In computing the likelihood $P(O)$ of an observation we need to sum up the probability of all paths in a Markov model. Brute force computation is not applicable in most cases. The forward algorithm is an application of dynamic programming.
The forward algorithm: estimation

Figure 6.6  The forward trellis for computing the total observation likelihood for the ice-cream events 3 / 3. Hidden states are in circles, observations in squares. White (unfilled) circles indicate illegal transitions. The figure shows the computation of $\alpha_t(j)$ for two states at two time steps. The computation in each cell follows Eq’ 6.11: $\alpha_t(j) = \sum_{i=1}^{N-1} \alpha_{t-1}(i) a_{ij} b_j(o_t)$. The resulting probability expressed in each cell is Eq’ 6.10: $\alpha_t(j) = P(o_1, o_2 \ldots o_t, q_t = j | \lambda)$. 
HMM Decoding: Viterbi Algorithm

Intuition:
Figure 6.9 The Viterbi trellis for computing the best path through the hidden state space for the ice-cream eating events 3 1 3. Hidden states are in circles, observations in squares. White (unfilled) circles indicate illegal transitions. The figure shows the computation of $v_t(j)$ for two states at two time steps. The computation in each cell follows Eq. 6.10: $v_t(j) = \max_{1 \leq i < N-1} v_{t-1}(i) \alpha_t b_j(\alpha_t)$ The resulting probability expressed in each cell is Eq. 6.16: $v_t(j) = P(q_0, q_1, \ldots, q_{t-1}, q_t | o_1, o_2, \ldots, o_t, q_t = j| \lambda)$. 
Viterbi decoding

function VITERBI(observations of len $T$, state-graph) returns best-path

$num\text{-states} \leftarrow \text{NUM-OF-STATES}(\text{state-graph})$
Create a path probability matrix $viterbi[num\text{-states}+2,T+2]$
$viterbi[0,0] \leftarrow 1.0$

for each time step $t$ from 1 to $T$ do
  for each state $s$ from 1 to num-states do
    $viterbi[s,t] \leftarrow \max_{1 \leq s' \leq \text{num-states}} viterbi[s',t-1] \ast a_{s',s} \ast b_s(o_t)$
    $\text{backpointer}[s,t] \leftarrow \arg\max_{1 \leq s' \leq \text{num-states}} viterbi[s',t-1] \ast a_{s',s}$
  Backtrace from highest probability state in final column of $viterbi[]$ and return path

Figure 6.10 Viterbi algorithm for finding optimal sequence of tags. Given an observation sequence and an HMM $\lambda = (A,B)$, the algorithm returns the state-path through the HMM which assigns maximum likelihood to the observation sequence. Note that states 0 and N+1 are non-emitting start and end states.
NLP and HMM decoding

- The HMM sequence labeling approach can be applied to a variety of linguistic subtasks:
  - Tokenization
  - MWE recognition
  - POS tagging
  - Named Entity Recognition
  - Predicate Argument Structure Recognition
  - SRL: Shallow Semantic Parsing
Multiword Expressions

he was willing to budge a little on

O O O O B b i l

the price which means a lot to me .

O O O B i l l l l O

a little; means a lot to me; budge . . . on

See: “Discriminative lexical semantic segmentation with gaps: running the MWE gamut,” Schneider et al. (2014).
With Commander Chris Ferguson at the helm,

Atlantis touched down at Kennedy Space Center.
Part-of-Speech Tagging

i k r  s m h  h e  a s k e d  f i r  y o  l a s t  n a m e
!  G  O  V  P  D  A  N
interjection  acronym  pronoun  verb  prep.  det.  adj.  noun

s o  h e  c a n  a d d  u  o n  f b  l o l o l o l l
P  O  V  V  O  P  ^  !
preposition  proper noun
Supersense Tagging

ikr  smh  he    asked  fir  yo  last  name
         communication         cognition

so  he  can  add  u  on  fb  lololol
         stative           group

NLP & Structured Prediction

- HMM Decoding is an example of a large class of structured prediction task

- Key elements:
  - Transform a NLP task into a sequence of classification problem.
  - Transform into a sequence labeling problem and use a variant of the Viterbi algorithm.
  - Design a representation (e.g. features and metrics), a prediction algorithm, and a learning algorithm for your particular problem.
Discriminative Learning

• Characterizes neural networks since the early Cybernetics (Minsky&Papert, 1956)

• Strongly rooted in the notion of
  • Inner product that in turns characterizes the norms thus the distances in the space

  • Use a vector space in $R^n$ as a input representation space

• (not so) Recent Achievements
  • Statistical Learning Theory and Support Vector Machines (Vapnik, 1987)
  • Deep Learning (Bengio et al., 2001)
**Linear Classification (1)**

In the hyperplane equation:

\[ f(x) = \bar{x} \cdot \bar{w} + b, \quad \bar{x}, \bar{w} \in \mathbb{R}^n, b \in \mathbb{R} \]

\( \bar{x} \) is the vector describing the targeted input example

\( \bar{w} \) is the gradient of the hyperplane

Classification Inference: \( h(x) = \text{sign}(f(x)) \)
Support Vector Machines

- Support Vector Machines (SVMs) are based on the Statistical Learning Theory [Vapnik, 1995]
  - It does not require the storage of all data but only a subset of discriminating instances (i.e. the support vectors, SV)
  - The classifier is a linear combination of the SVs (i.e. it depends ONLY on the inner product with their vectors)

\[ h(x) = \text{sgn}(\vec{w} \cdot \vec{x} + b) = \text{sgn}\left( \sum_{j=1}^{\ell} \alpha_j y_j \vec{x}_j \cdot \vec{x} + b \right) \]
Separability in Higher Dimensional spaces

- In $\mathbb{R}^2$, 3 points can always be separated (or shuttered) by a linear classifier but
  - 4 punti do not (as VC=3) [Vapnik and Chervonenkis(1971)]

- Solution 1 (neural networks): complexify the classification function
  - It needs a more complex architecture usually based on *ensembles* (ad es. multistrato) of neurons
  - Risk of over-fitting on the training data that is dangerous for performance on test ones
Separability and High Dimensional Spaces (2)

- Solution 2: Project instances in an higher dimensional space, i.e. a new feature space by using a projection function $\phi$

- Basic idea from SLT: Feature space more complex are preferable to more complex functions as the risk of overfitting is minimized
• If a specific function called kernel is available such that \( k(x_i, x_j) = \phi(x_i) \cdot \phi(x_j) \), there is no need to project the individual examples through the projection function \( \phi \) (Cristianini et al., 2002)

• A structured paradigm is applied such that
  • It is trained against more complex structures
  • It moves the machine learning focus onto the representation \( (\phi(x_j)) \)

• \( k(.,.) \) expresses a similarity (metrics) that can account for linguistic aspects and depend on the **lexicon**, **syntax** and/or **semantics**

\[
h(x) = \text{sgn}(\tilde{w} \cdot \varphi(\tilde{x}) + b) = \text{sgn}(\sum_{j=1}^{\ell} \alpha_j y_j \varphi(\tilde{x}_j) \cdot \varphi(\tilde{x}) + b) = \\
= \text{sgn}(\sum_{i=1}^{\ell} \alpha_j y_j k(\tilde{x}_j, \tilde{x}) + b)
\]
Examples of Kernels sensitive to syntactic structures

- Given a tree we can see it as the occurrence of a joint event.
Kernels & Syntactic structures: a collective view of the joint event

The tree can be see it as the joint occurrence of all the following subtrees:
The function $\phi$ in a tree kernel define a vector representing ALL subtrees of the input tree $T$. It naturally (i.e. without feature engineering) emphasizes:

- Lexical information (*magazine*)
- Coarse grain grammatical information (POS tags such as *VBZ*)
- Syntactic information (frammenti complessi)

The inner product in the space of all subtrees is proportional to the number of subtrees shared between two sentences.

The learning algorithm (e.g. SVM) will select discriminating examples in (infinite dimensional) space.
Application of distributional lexicons for Semantic Role Labeling @ UTV

• An important application of tree-kernel based SVMs is Semantic Role labeling wrt Framenet

• In the UTV system, a cascade of classification steps is applied:
  • Predicate detection
  • Boundary recognition
  • Argument categorization (Local models)
  • Reranking (Joint models)

• Input: a sentence and its parse trees

• Adopted kernel: the combination of lexical (e.g. bow) and tree kernel (that is still a kernel)
• *Police arrested the man for shoplifting*
Using Framenet/PropBank

SRL Pipeline

Syntactic Parse
- S
- NP₁
- VP
- V: Walked
- PP
- P: in
- NP₂: the park

Prune Constituents
- NP₁
- VP
- V
- PP
- NP₂

Argument Identification
- Argument
- NP₁ Yes
- VP No
- V given
- PP Yes
- NP₂ No

Argument Classification

Structural Inference
- NP₁ Agent
- V: Predicate
- PP: Location
- Semantic Roles

Candidates
- NP₁ Agent/Patient
- V: Predicate
- PP: Location/Patient
 Semantic Role Labeling via SVM Learning

• Two steps:
  • Boundary Detection
    • One binary classifier applied to the parse tree nodes
  • Argument Type Classification
    • Multi-classification problem, where n binary classifiers are applied, one for each argument class (i.e. frame element)
    • They are combined in a ONE-vs-ALL scheme, i.e. the argument type that is categorized by an SVM with the maximum score is selected
SRL in Framenet: Results

<table>
<thead>
<tr>
<th>Eval Setting</th>
<th>Tree Kernels</th>
<th>Tree Kernels + PK</th>
<th>PK alone</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$P$</td>
<td>$R$</td>
<td>$F_1$</td>
</tr>
<tr>
<td>BD</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BD Proj.</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BD+RC</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BD+RC Proj.</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>TKL</td>
<td>TK</td>
<td>TK + PK</td>
<td></td>
</tr>
<tr>
<td>BD</td>
<td>.949</td>
<td>.652</td>
<td>.773</td>
</tr>
<tr>
<td>BD Proj.</td>
<td>.919</td>
<td>.631</td>
<td>.748</td>
</tr>
<tr>
<td>BD+RC</td>
<td>.697</td>
<td>.479</td>
<td>.568</td>
</tr>
<tr>
<td>BD+RC Proj.</td>
<td>.672</td>
<td>.462</td>
<td>.548</td>
</tr>
</tbody>
</table>

Table 4.1: Results on FrameNet dataset. The table shows Precision, Recall, and F-measure achieved by the Polynomial Kernel (PK) and two different Tree Kernels (TK and TKL). Also, results for their combinations are shown. All experiments exploit 2% training data for Boundary Detection, and 90% for Role Classification.
Framenet SRL: best results

- Best system [Erk&Pado, 2006]
  - 0.855 Precision, 0.669 Recall
  - 0.751 F1
- Trento (+RTV) system (Coppola, PhD2009)

<table>
<thead>
<tr>
<th>Enhanced PK+TK</th>
<th>P</th>
<th>R</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>BD (nodes)</td>
<td>1.0</td>
<td>.732</td>
<td>.847</td>
</tr>
<tr>
<td>BD (words)</td>
<td>.963</td>
<td>.702</td>
<td>.813</td>
</tr>
<tr>
<td>BD+RC (nodes)</td>
<td>.784</td>
<td>.571</td>
<td>.661</td>
</tr>
<tr>
<td>BD+RC (words)</td>
<td>.747</td>
<td>.545</td>
<td>.630</td>
</tr>
</tbody>
</table>

Table 4.2: Results on the FrameNet dataset. Best configuration from Table 4.1, raised to 90% of training data for BD and RC.
Argument Classification (Croce et al., 2013)

- UTV experimented with a FrameNet SRL classification (gold standard boundaries)
- We used the FrameNet version 1.3: 648 frames are considered
  - Training set: 271,560 arguments (90%)
  - Test set: 30,173 arguments (10%)

\[ \text{[Bootleggers]}_{\text{CREATOR}}, \text{then copy [the film]}_{\text{ORIGINAL}} \text{[onto hundreds of VHS tapes]}_{\text{GOAL}} \]

<table>
<thead>
<tr>
<th>Kernel</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRCT</td>
<td>87.60%</td>
</tr>
<tr>
<td>GRCT\text{_SA}</td>
<td>88.61%</td>
</tr>
<tr>
<td>LCT</td>
<td>87.61%</td>
</tr>
<tr>
<td>LCT\text{_SA}</td>
<td>88.74%</td>
</tr>
<tr>
<td>GRCT+LCT</td>
<td>87.99%</td>
</tr>
<tr>
<td>GRCT\text{_SA}+LCT\text{_SA}</td>
<td>88.91%</td>
</tr>
</tbody>
</table>
Semantics, Natural Language & Learning

• From Learning to Read to Knowledge Distillation as a (integrated pool of) Semantic interpretation Task(s)
  • Information Extraction
    • Entity Recognition and Classification
    • Relation Extraction
    • Semantic Role Labeling (Shallow Semantic Parsing)
  • Estimation of Text Similarity
    • Structured Text Similarity/Textual Entailment Recognition
    • Sense disambiguation
  • Semantic Search, Question Classification and Answer Ranking
  • Knowledge Acquisition, e.g. ontology learning
  • Social Network Analysis, Opinion Mining
References


• NLP & ML:

• URLs:
  • SAG, Univ. Roma Tor Vergata: http://sag.art.uniroma2.it/
  • Reveal s.r.l.: http://www.revealsrl.it/