A Markovian Kernel-based Approach for itaLIan Speech acT labEliNg

Danilo Croce and Roberto Basili
University of Roma, Tor Vergata

ILISTEN@EVALITA – Torino 2018

Introduction

- In this talk, the UNITOR system participating in the iLISTEN@EvalIta 2018 is presented
  - **Task**: given a list of sentences within a dialogue, to assign each turn to a speech act class
- It seems a "standard" sentence classification task…

  … but turns are not observed in isolation, they belong to the dialogue
- UNITOR is essentially a Markovian classifier
  - the classification of the i^{th} utterance also depends from the dialogue act assigned at the previous utterance.

If your are interested in Deep Methods…

Markovian SVM

- **AIM**: to make the classification of an example $x_i \in R^n$ (from a sequence) dependent on the label assigned to the previous elements
  - Within in a history of length m, i.e., $x_{i-m}, \ldots, x_{i-1}$.
- A dialogue is a sequence of utterances $x = (x_1, \ldots, x_s)$ each of them representing the specific i^{th} utterance.
- Given the corresponding sequence of expected labels $y = (y_1, \ldots, y_s)$, a sequence of m step-specific labels can be retrieved, in the form $y_{i-m}, \ldots, y_{i-1}$.
- **IDEA**: to augment the feature vector of x_i with a projection function $\psi_m(x_i) \in R^{md}$
  - We augment x_i with features indicating one of the possible labels observed in a history of length m

(slightly) More formally

- Given the SVM, a projection function $\varphi_m(\cdot)$ can be defined to consider both x_i and the transitions $\psi_m(x_i)$ by concatenating the two representations:

$$\varphi_m(x_i) = x_i \,\|\, \psi_m(x_i)$$

- **Kernel-based methods** can be applied:
  - the feature representing individual turns
  - the information about the transitions within the dialogue.
- We define a kernel function between turns surrogating the product between $\varphi_m(\cdot)$ such that:

$$K_m(x_i, z_j) = \varphi_m(x_i)\,\varphi_m(z_j) = \boxed{K^{obs}(x_i, z_j)} + \boxed{K^{tr}(\psi_m(x_i), \psi_m(z_j))}$$

It does not depend on m

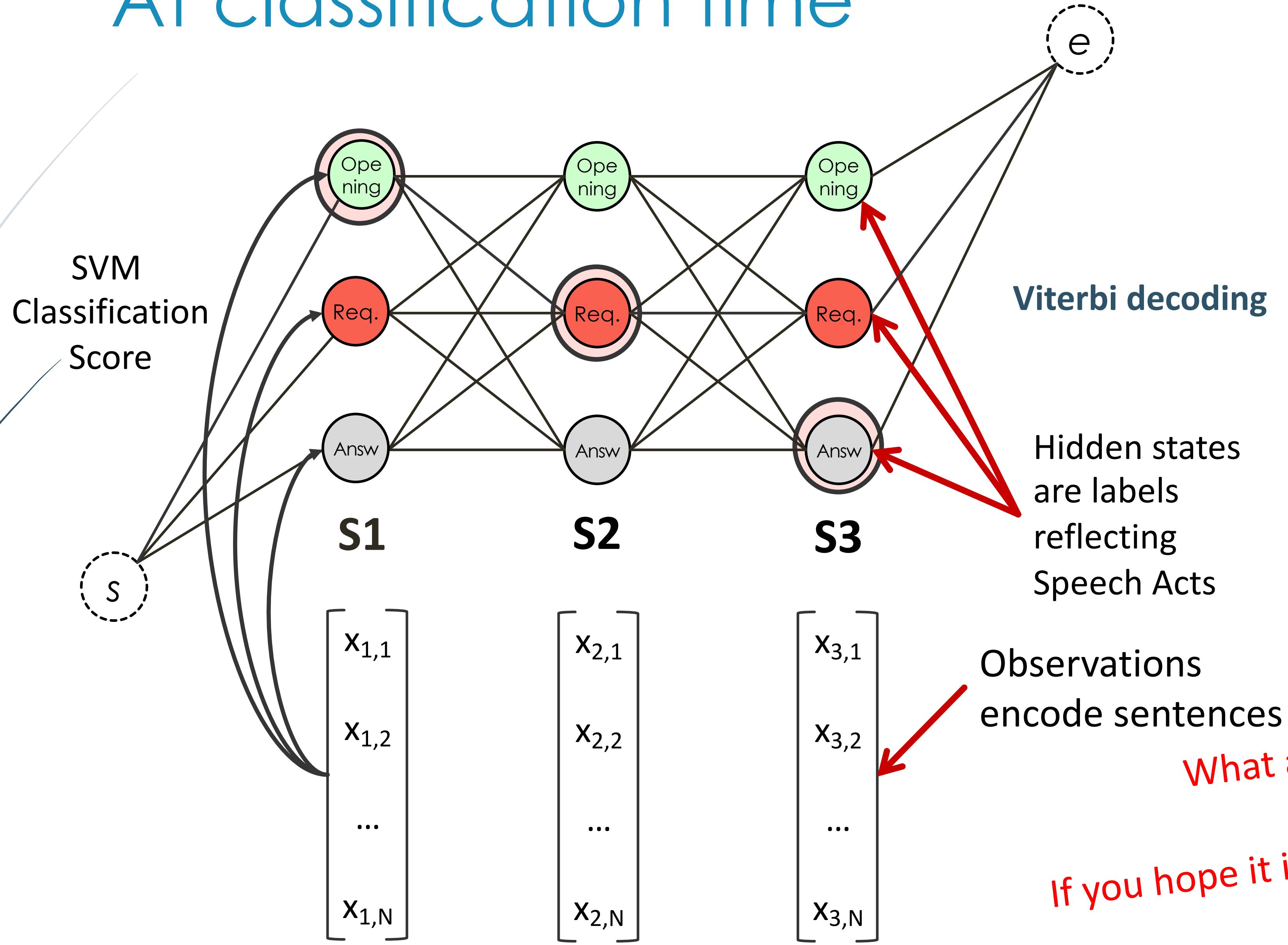
A Markovian Kernel-based Approach

- If we define K^{tr} as a linear kernel between input instances, i.e. a dot-product in the space generated by $\psi_m(\cdot)$:

$$K_m(x_i, z_j) = K^{obs}(x_i, x_j) + \psi_m(x_i)\,\psi_m(z_j)$$

- At **training time** we just use the kernel-based SVM in a One-Vs-All schema over the feature space derived by $K_m(\cdot, \cdot)$

At classification time



What about K^{obs} ?

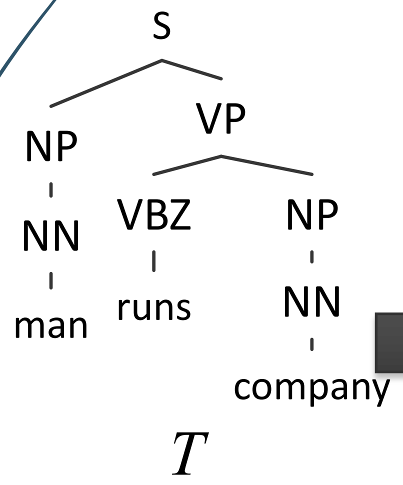
If you hope it is not a tree kernel…

Structural kernels

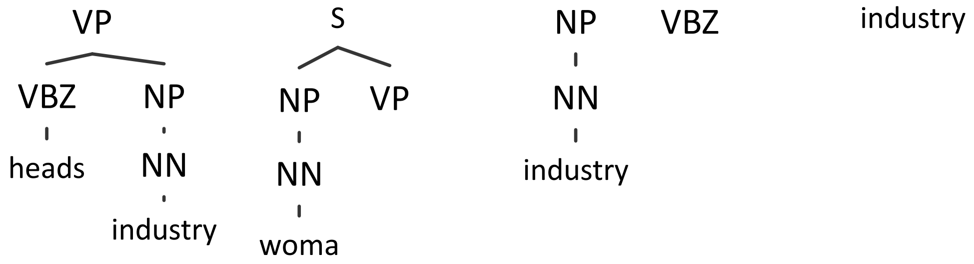
- Convolution Kernels systematically account for structural analogies/similarities between discrete structures
- Tree Kernels account for structural analogies between syntactic parse trees
  - They express **the number of the shared substructures between two syntactic trees**

Tree Kernels: the IDEA

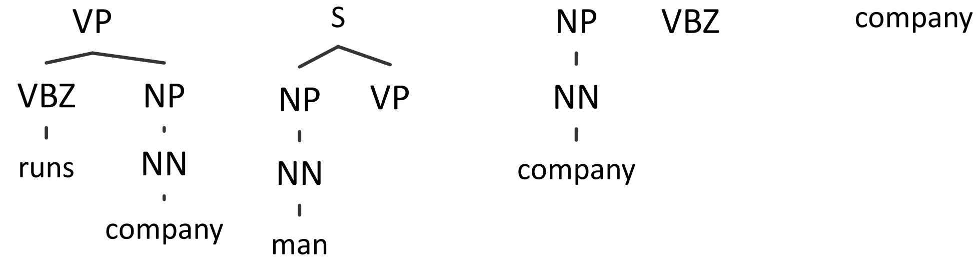
- Tree Kernels account for structural analogies between syntactic parse trees
- **Smoothed Partial Tree Kernels** (SPTKs) jointly model **syntactic** and **lexical semantic** similarity within Kernel functions



$$\phi(T_1) = (0,\ldots,1,\ldots,0,\ldots,1,\ldots,0,\ldots,1,\ldots,1,\ldots,0,\ldots,1)$$



$$\phi(T_2) = (0,\ldots,1,\ldots,0,\ldots,1,\ldots,0,\ldots,1,\ldots,1,\ldots,0,\ldots,1)$$



SPTK: Formal definition

- Given two trees T_1 and T_2
  - If n_1 and n_2 are leaves then

$$\Delta_\sigma(n_1, n_2) = \mu\lambda\sigma_\tau(n_1, n_2)$$

$$\Delta_\sigma(n_1, n_2) = \mu\sigma_\tau(n_1, n_2) \times \Big(\lambda^2 + \sum_{\vec{I}_1, \vec{I}_2, l(\vec{I}_1)=l(\vec{I}_2)} \lambda^{d(\vec{I}_1)+d(\vec{I}_2)} \prod_{j=1}^{l(\vec{I}_1)} \Delta_\sigma(c_{n_1}(\vec{I}_{1j}), c_{n_2}(\vec{I}_{2j}))\Big)$$

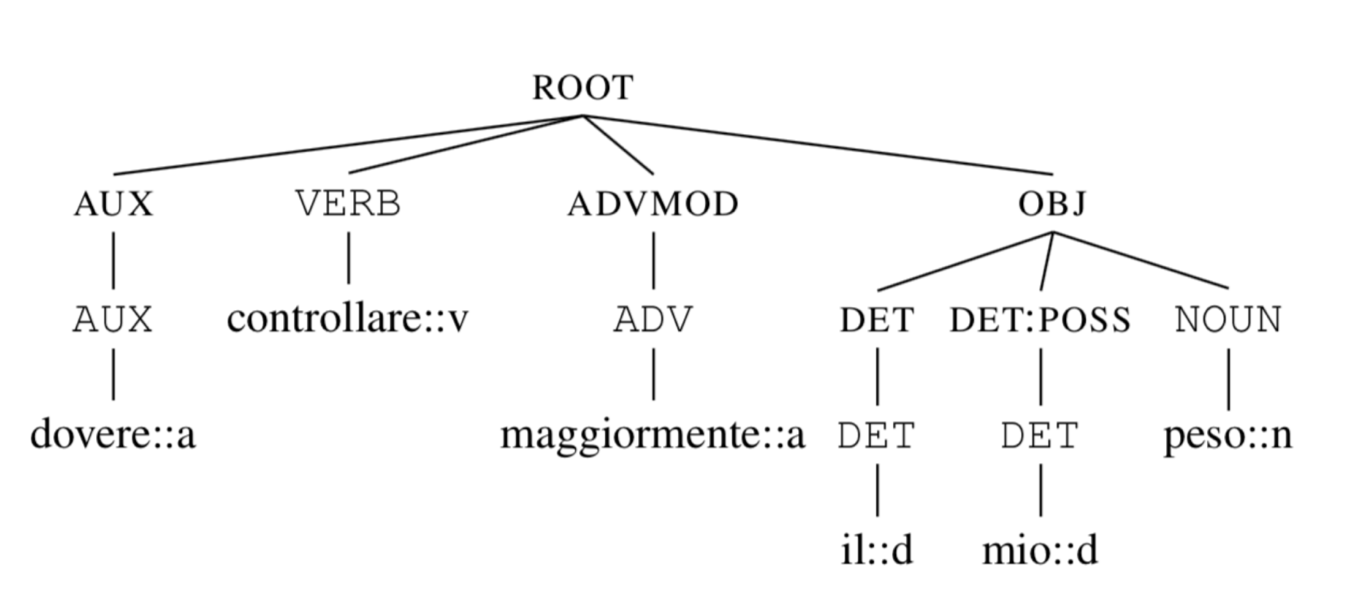
$\sigma(n_1, n_2)$ is a similarity function among the tree nodes depending on their linguistic type τ

Algorithm 1 $\sigma_\tau(n_1, n_2, lw)$

$\sigma_\tau \leftarrow 0,$
if $\tau(n_1) = \tau(n_2) = \text{SYNT} \wedge label(n_1) = label(n_2)$ **then**
  $\sigma_\tau \leftarrow 1$
end if
if $\tau(n_1) = \tau(n_2) = \text{POS} \wedge label(n_1) = label(n_2)$ **then**
  $\sigma_\tau \leftarrow 1$
end if
if $\tau(n_1) = \tau(n_2) = \text{LEX} \wedge pos(n_1) = pos(n_2)$ **then**
  $\sigma_\tau \leftarrow \sigma_{LEX}(n_1, n_2)$
end if
if leaf$(n_1) \wedge$ leaf(n_2) **then**
  $\sigma_\tau \leftarrow \sigma_\tau \times lw$
end if
return σ_τ

From sentences to trees

- The SPTK is applied to trees derived from the dependency parse tree
  - The structure encodes syntactic and semantic information
  - No task specific feature engineering



Results

- The dataset was syntactically parsed with SpaCy
- We used the Kernel-based SVM-HMM implemented in KeLP
- Parameters has been tuned according to a n-fold cross validation schema
  - An history of $m = 1$ is adopted

Run	Micro			Macro		
	P	**R**	**F1**	**P**	**R**	**F1**
UNITOR	.733	.733	**.733**	.681	.628	**.653**
System2	.685	.685	.685	.608	.584	.596
Baseline	.340	.340	.340	.037	.111	.056

Conclusions

- The proposed classification strategy shows the beneficial impact of
  - a structured kernel-based method
  - with a Markovian classifier,
- It seems capitalizing the contribution of the dialogue modeling in deciding the speech act of individual sentences.
  - No requirement in term of task-specific feature and system engineering
  - Results are appealing mostly considering the reduced size of the dataset
- Further work: combination of the adopted strategy with recurrent neural approaches.