A Markovian Kernel-based Approach for itaLIan Speech acT labEliNg

Danilo Croce and **Roberto Basili**
University of Roma, Tor Vergata
Via del Politecnico 1, Rome, 00133, Italy
{croce,basili}@info.uniroma2.it

Abstract

English. This paper describes the UNITOR system that participated to the *itaLIan Speech acT labEliNg* task within the context of EvalIta 2018. A Structured Kernel-based Support Vector Machine has been here applied to make the classification of the dialogue turns sensitive to the syntactic and semantic information of each utterance, without relying on any task-specific manual feature engineering. Moreover, a specific Markovian formulation of the SVM is adopted, so that the labeling of each utterance depends on speech acts assigned to the previous turns. The UNITOR system ranked first in the competition, suggesting that the combination of the adopted structured kernel and the Markovian modeling is beneficial.

Italian. *Questo lavoro descrive il sistema UNITOR che ha partecipato all'itaLIan Speech acT labEliNg task organizzato nell'ambito di EvalIta 2018. Il sistema è basato su una Structured Kernel-based Support Vector Machine (SVM) che rende la classificazione dei turni di dialogo dipendente dalle informazioni sintattiche e semantiche della frase, evitando la progettazione di alcuna feature specifica per il task. Una specifica formulazione Markoviana dell'algoritmo di apprendimento SVM permette inoltre di etichettare ciascun turno in funzione delle classificazioni dei turni precedenti. Il sistema UNITOR si é classificato al primo posto nella competizione, e questo conferma come la combinazione della funzione kernel e del modello Markoviano adottati sia molto utile allo sviluppo di sistemi di dialoghi robusti.*

1 Introduction

A dialogue agent is designed to interact and communicate with other agents, in a coherent manner, not just through one-shot messages, but according to sequences of meaningful and related messages on an underlying topic or in support to an overall goal (Traum, 1999). These communications are seen not just as transmitting information but as actions that change the state of the world, e.g., the mental states of the agents involved in the conversation, as well as the state, or context, of the dialogue. In other words, speech act theory allows to design an agent in order to place its communication within the same general framework as the agent's actions. In such a context, the robust recognition of the speech acts characterizing an interaction is crucial for the design and deployment of artificial dialogue agents.

This specific task has been considered in the first *itaLIan Speech acT labEliNg task at EvalIta* (iLISTEN, (Novielli and Basile, 2018)): given a dialogue between an agent and a user, the task consists in automatically annotating the user's dialogue turns with speech act labels, i.e. with the communicative intention of the speaker, such as *statement*, *request for information* or *agreement*. Table 1 reports the full set of speech act labels considered in the challenge, with definition and examples.

In this paper, the UNITOR system participating in the iLISTEN task within the EvalIta 2018 evaluation campaign is described. The system realize the classification task through a Structured Kernel-based Support Vector Machine (Vapnik, 1998) classifier. A structured kernel, namely a Smoothed Partial Tree Kernel (SPTK, (Croce et al., 2011)) is applied in order to make the classification of each utterance dependent from the syntactic and semantic information of each individual utterance.

Since turns are not observed in isolation, but immersed in a dialogue, we adopted a Markovian for-

Speech Act	Description	Example
OPENING	Dialogue opening or self-introduction	*Ciao, io sono Antonella*
CLOSING	Dialogue closing, e.g. farewell, wishes, intention to close the conversation	*Va bene, ci vediamo prossimamente*
INFO-REQUEST	Utterances that are pragmatically, semantically, and syntactically questions	*E cosa mi dici delle vitamine?*
SOLICIT-REQ-CLARIF	Request for clarification (please explain) or solicitation of system reaction	*Mmm, si ma in che senso?*
STATEMENT	Descriptive, narrative, personal statements	*Devo controllare maggiormente il mio peso.*
GENERIC-ANSWER	Generic answer	*Si, No, Non so*
AGREE	Expression of agreement, e.g. acceptance of a proposal, plan or opinion	*Si, so che importante.*
REJECT	Expression of disagreement, e.g. rejection of a proposal, plan, or opinion	*Ho sentito tesi contrastanti al proposito.*
KIND-ATT-SMALLTALK	Expression of kind attitude through politeness, e.g. thanking, apologizing or smalltalk	*Grazie,Sei per caso offesa per qualcosa che ho detto?*

Table 1: Full set of speech act labels considered at iLISTEN

mulation of SVM, namely SVM^{hmm} (Altun et al., 2003) so that the classification of the i^{th} utterance also depends from the dialogue act assigned at the $i-1^{th}$ utterance.

The UNITOR system ranked first in the competition, suggesting that the combination of the adopted structured kernel and the Markovian learning algorithm is beneficial.

In the rest of the paper, Section 2 describes the adopted machine learning method and the underlying semantic kernel functions. In Section 3, the performance measures of the system are reported while Section 4 derives the conclusions.

2 A Markovian Kernel-based Approach

The UNITOR system implements a Markovian formulation of the Support Vector Machine (SVM) learning algorithm. The SVM adopts a structured kernel function in order to estimate the syntactic and semantic similarity between utterances in a dialogue, without the need of any task-specific manual feature engineering (only the dependency parse of each sentence is required). In the rest of this section, first the learning algorithm is presented, then the adopted kernel method is discussed.

2.1 A Markovian Support Vector Machine

The aim of a Markovian formulation of SVM is to make the classification of a input example $x_i \in \mathbb{R}^n$ (belonging to a sequence of examples) dependent on the label assigned to the previous elements in a history of length m, i.e., $x_{i-m}, \ldots, x_{i-1}$.

In our classification task, a dialogue is a sequence of utterances $\mathbf{x} = (x_1, \ldots, x_s)$ each of them representing an example x_i, i.e., the specific i-th utterance. Given the corresponding sequence of expected labels $\mathbf{y} = (y_1, \ldots, y_s)$, a sequence of m step-specific labels (from a dictionary of d different dialogue acts) can be retrieved, in the form $y_{i-m}, \ldots, y_{i-1}$.

In order to make the classification of x_i dependent also from the previous decisions, we augment the feature vector of x_i introducing a projection function $\psi_m(x_i) \in \mathbb{R}^{md}$ that associates to each example a $md-$dimensional feature vector where each dimension set to 1 corresponds to the presence of one of the d possible labels observed in a history of length m, i.e. m steps before the target element x_i.

In order to apply a SVM, a projection function $\phi_m(\cdot)$ can be defined to consider both the observations x_i and the transitions $\psi_m(x_i)$ by concatenating the two representations as follows:

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

with $\phi_m(x_i) \in \mathbb{R}^{n+md}$. Notice that the symbol $||$ here denotes the vector concatenation, so that $\psi_m(x_i)$ does not interfere with the original feature space, where x_i lies.

Kernel-based methods can be applied in order to model meaningful representation spaces, encoding both the feature representing individual examples together with the information about the transitions. According to kernel-based learning (Shawe-Taylor and Cristianini, 2004), we can define a kernel function $K_m(x_i, z_j)$ between a generic item of a sequence x_i and another generic item z_j from the same or a different sequence, parametric in the history length m: it surrogates the product between

$\phi_m(\cdot)$ such that:

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

In other words, we define a kernel that is the linear combination of two further kernels: K^{obs} operating over the individual examples x_i and a K^{tr} operating over the feature vectors encoding the involved transitions. It is worth noticing that K^{obs} does not depend on the position nor the context of individual examples in line with Markov assumption characterizing a large class of these generative models, e.g. HMM. For simplicity, 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 use the kernel-based SVM in a One-Vs-All schema over the feature space derived by $K_m(\cdot,\cdot)$. The learning process provides a family of classification functions $f(x_i; m) \subset \mathbb{R}^{n+md} \times \mathbb{R}^d$, which associate each x_i to a distribution of scores with respect to the different d labels, depending on the context size m. At classification time, all possible sequences $\mathbf{y} \in \mathcal{Y}^+$ should be considered in order to determine the best labeling $\hat{\mathbf{y}}$, where m is the size of the history used to enrich x_i, that is:

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}\in\mathcal{Y}^+}\{\sum_{i=1\ldots m} f(x_i; m)\}$$

In order to reduce the computational cost, a *Viterbi-like decoding algorithm* is adopted[1]. The next section defines the kernel function K^{obs} applied to specific utterances.

2.2 Structured Kernel Methods for Speech Act Labeling

Several NLP tasks require the explorations of complex semantic and syntactic phenomena. For instance, in Paraphrase Detection, verifying whether two sentences are valid paraphrases involves the analysis of some rewriting rules in which the syntax plays a fundamental role. In Question Answering, the syntactic information is crucial, as largely demonstrated in (Croce et al., 2011).

[1]When applying $f(x_i; m)$ the classification scores are normalized through a softmax function and probability scores are derived.

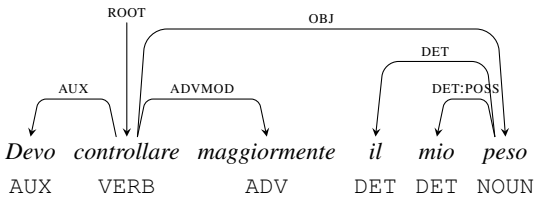

Figure 1: Dependency Parse Tree of *"Devo controllare maggiormente il mio peso"* (In English: *"I need to control my weight more"*)

A natural approach to exploit such linguistic information consists in applying kernel methods (Robert Müller et al., 2001; Shawe-Taylor and Cristianini, 2004) on structured representations of data objects, e.g., documents. A sentence s can be represented as a parse tree[2] that expresses the grammatical relations implied by s. Tree kernels (TKs) (Collins and Duffy, 2001) can be employed to directly operate on such parse trees, evaluating the tree fragments shared by the input trees. This operation corresponds to a dot product in the implicit feature space of all possible tree fragments.

Whenever the dot product is available in the implicit feature space, kernel-based learning algorithms, such as SVMs, can operate in order to automatically generate robust prediction models. TKs thus allow to estimate the similarity among texts, directly from sentence syntactic structures, that can be represented by parse trees. The underlying idea is that the similarity between two trees T_1 and T_2 can be derived from the number of shared tree fragments. Let the set $\mathcal{T} = \{t_1, t_2, \ldots, t_{|\mathcal{T}|}\}$ be the space of all the possible substructures and $\chi_i(n_2)$ be an indicator function that is equal to 1 if the target t_i is rooted at the node n_2 and 0 otherwise. A tree-kernel function over T_1 and T_2 is defined as follows: $TK(T_1, T_2) = \sum_{n_1\in N_{T_1}} \sum_{n_2\in N_{T_2}} \Delta(n_1, n_2)$ where N_{T_1} and N_{T_2} are the sets of nodes of T_1 and T_2 respectively, and $\Delta(n_1, n_2) = \sum_{k=1}^{|\mathcal{T}|} \chi_k(n_1)\chi_k(n_2)$ which computes the number of common fragments between trees rooted at nodes n_1 and n_2. The feature space generated by the structural kernels obviously depends on the input structures. Notice that different tree representations embody different linguistic theories and may produce more or less effective syntactic/semantic feature spaces for a

[2]Parse trees can be extracted using automatic parsers. In our experiments, we used SpaCy `https://spacy.io/`.

given task.

Dependency grammars produce a significantly different representation which is exemplified in Figure 1. Since tree kernels are not tailored to model the labeled edges that are typical of dependency graphs, these latter are rewritten into explicit hierarchical representations. Different rewriting strategies are possible, as discussed in (Croce et al., 2011): a representation that is shown to be effective in several tasks is the Grammatical Relation Centered Tree (GRCT) illustrated in Figure 2: the PoS-Tags are children of grammatical function nodes and direct ancestors of their associated lexical items.

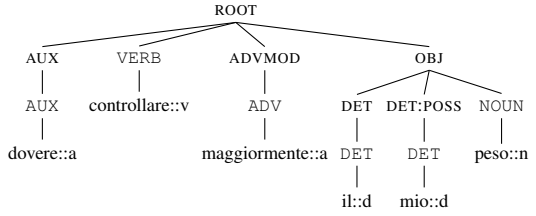

Figure 2: Grammatical Relation Centered Tree (GRCT) of *"Devo controllare maggiorment il mio peso"*.

Different tree kernels can be defined according to the types of tree fragments considered in the evaluation of the matching structures. In the *Subtree Kernel* (Vishwanathan and Smola, 2002), valid fragments are only the grammatically well formed and complete subtrees: every node in a subtree corresponds to a context free rule whose left hand side is the node label and the right hand side is completely described by the node descendants. Subset trees are exploited by the *Subset Tree Kernel* (Collins and Duffy, 2001), which is usually referred to as Syntactic Tree Kernel (STK); they are more general structures since their leaves can be non-terminal symbols. The subset trees satisfy the constraint that grammatical rules cannot be broken and every tree exhaustively represents a CFG rule. *Partial Tree Kernel* (PTK) (Moschitti, 2006) relaxes this constraint considering partial trees, i.e., fragments generated by the application of partial production rules (e.g. sequences of non terminal with gaps). The strict constraint imposed by the STK may be problematic especially when the training dataset is small and only few syntactic tree configurations can be observed. The Partial Tree Kernel (PTK) overcomes this limitation, and usually leads to higher accuracy, as shown in (Moschitti, 2006).

Capitalizing lexical information in Convolution Kernels. The tree kernels introduced above perform a hard match between nodes when comparing two substructures. In NLP tasks, when nodes are words, this strict requirement reflects in a too strict lexical constraint, that poorly reflects semantic phenomena, such as the synonymy of different words or the polysemy of a lexical entry. To overcome this limitation, we adopt Distributional models of Lexical Semantics (Sahlgren, 2006; Mikolov et al., 2013) to generalize the meaning of individual words by replacing them with geometrical representations (also called Word Embeddings) that are automatically derived from the analysis of large-scale corpora. These representations are based on the idea that words occurring in the same contexts tend to have similar meaning: the adopted distributional models generate vectors that are similar when the associated words exhibit a similar usage in large-scale document collections. Under this perspective, the distance between vectors reflects semantic relations between the represented words, such as paradigmatic relations, e.g., quasi-synonymy. These word spaces allow to define meaningful soft matching between lexical nodes, in terms of the distance between their representative vectors. As a result, it is possible to obtain more informative kernel functions which are able to capture syntactic and semantic phenomena through grammatical and lexical constraints.

The *Smoothed Partial Tree Kernel* (SPTK) described in (Croce et al., 2011) exploits this idea extending the PTK formulation with a similarity function σ between nodes:

$$\Delta_{SPTK}(n_1, n_2) = \mu\lambda\sigma(n_1, n_2)\ , \text{ if } n_1 \text{ and } n_2 \text{ are leaves}$$
$$\Delta_{SPTK}(n_1, n_2) = \mu\sigma(n_1, n_2)\Big(\lambda^2 + \sum_{\vec{I}_1, \vec{I}_2 : l(\vec{I}_1) = l(\vec{I}_2)} \lambda^{d(\vec{I}_1)+d(\vec{I}_2)} \prod_{k=1}^{l(\vec{I}_1)} \Delta_{SPTK}\left(c_{n_1}(i_k^1), c_{n_2}(i_k^2)\right)\Big) \quad (1)$$

In the SPTK formulation, the similarity function $\sigma(n_1, n_2)$ between two nodes n_1 and n_2 can be defined as follows:

- if n_1 and n_2 are both lexical nodes, then $\sigma(n_1, n_2) = \sigma_{LEX}(n_1, n_2) = \tau \frac{\vec{v}_{n_1} \cdot \vec{v}_{n_2}}{\left\|\vec{v}_{n_1}\right\| \left\|\vec{v}_{n_2}\right\|}$. It is the cosine similarity between the word vectors $\vec{v}_{n_1}$ and $\vec{v}_{n_2}$ associated with the labels of n_1 and n_2, respectively. τ is called *terminal factor* and weighs the contribution

of the lexical similarity to the overall kernel computation.

- else if n_1 and n_2 are nodes sharing the same label, then $\sigma(n_1, n_2) = 1$.
- else $\sigma(n_1, n_2) = 0$.

In the challenge we adopt the SPTK in order to implement the K^{obs} function used in the Markovian SVM. This kernel in fact has been showed very robust in the classification of (possibly short) sentences, such as in Question Classification (Croce et al., 2011) or Semantic Role Labeling (Croce et al., 2011; Croce et al., 2012).

Dataset	#dialogues	#user turns	#system turns	#total turns
train	40	1,097	1,119	2,216
test	20	479	492	971
complete	60	1,576	1,611	3,187

Table 2: Statistics about the iLISTEN dataset

3 Experimental Results

In iLISTEN, the reference dataset includes the transcriptions of 60 dialogues amounting to about $22,000$ words. The detailed statistics regarding dialogues and turns in the train and test dataset are reported in Table 2.

	Micro			Macro		
Run	**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

Table 3: Results of the UNITOR system

In the proposed classification workflow each utterance from the training/test material is processed through the SpaCy[3] dependency parser whose outputs are automatically converted into GRCT structures[4] discussed in Section 2. These structures are used within the Markovian SVM implemented in KeLP[5] (Filice et al., 2015; Filice et al., 2018).

The learning algorithm is based on a SPTK combined with a One-Vs-All multi-classification schema adopted to assign individual utterances to the targeted classes. All the parameters of the specific kernel (i.e. the contribution of the lexical nodes in the overall computation) and of the SVM algorithm have been tuned via 10-cross fold validation over the training set. In the Markovian SVM, a history of $m = 1$ previous steps allowed to achieve the best results during the parameterization step. Given the limited size of the dataset, higher values of m led to sparse representation of the transitions ψ_m that are not helpful. As a multi-classification task, results are reported in terms of precision (P), recall (R) and F1-score with respect to the gold standard, as reported in Table 3. These are averaged across each utterance (*micro*-statistics) and per class (*macro*-statistics).

Among the two submitted systems, UNITOR (reported on top of the table) achieved best results, both considering micro and macro statistics, where a F1 of 0.733 and 0.653 are achieved, respectively. These results are higher with respect to the other participant (namely System2) and far higher than the baseline (that confirms the difficulty of the task).

Given the unbalanced number of examples for each class, UNITOR achieves higher results w.r.t. the micro statistics, while lower results are achieved w.r.t. classes with a reduced number of examples. The confusion matrix reported in Table 4 shows that some recurrent misclassifications (e.g. the `STATEMENT` class with respect to the `REJECT` class) need to be carefully addressed. Clearly this is a very challenging task, also for the annotators, where the differences between the speech act is not strictly defined: as for example, given the stimulus of the system "*Bisognerebbe mangiare solo se si ha fame, ed aspettare che la digestione sia completata prima di assumere altri cibi*[6]" the answer "*a volte il lavoro non mi permette di mangiare con ritmi regolari!*[7]" should be labeled as `REJECT` while the system provides `STATEMENT`. Overall this results is straightforward, also considering that the system did not required any task specific feature modeling, but the adopted structured kernel based method allows capitalizing the syntactic and semantic information useful for the task. The only requirement of the system is the availability of a dependency parser.

[3] It is freely available for several languages (including Italian) at `https://spacy.io/`

[4] Utterances may include more than one sentence and potentially generate different trees. These cases are handled as follows: all trees after the first one are linked through the creation of an artificial link between their roots and the root of the tree generated by the first sentence.

[5] `http://www.kelp-ml.org/?page_id=215`

[6] Translated in English: "*You should only eat if you are hungry, and wait until digestion is complete before eating again.*"

[7] Translated in English: "*sometimes work doesn't allow me to eat at a regular pace!*"

	STATEMENT	KIND-ATT.	GEN.-ANSW.	REJECT	CLOSING	SOL.-CLAR.	OPENING	AGREE	INFO-REQ.
STATEMENT	**153**	6	3	24	0	3	0	2	13
KIND-ATT.	4	**17**	0	5	1	2	0	3	2
GEN.-ANSW.	1	0	**48**	0	0	1	0	6	0
REJECT	0	3	0	**3**	0	0	0	0	1
CLOSING	0	0	0	0	**7**	1	0	1	0
SOL.-CLAR.	0	6	0	2	1	**8**	0	1	2
OPENING	0	0	0	0	0	0	**11**	0	0
AGREE	0	3	1	1	0	0	0	**11**	1
INFO-REQ.	4	9	0	4	1	9	0	0	**93**

Table 4: Confusion Matrix of the UNITOR system w.r.t. gold standard. In column the number of classes from the gold standard, while rows report the system decisions. In bold correct classifications.

4 Conclusions

In this paper the description of the UNITOR system participating to the iLISTEN task at EvalIta 2018 has been provided. The system ranked first in all the evaluations. Thus, the proposed classification strategy shows the beneficial impact of the combination of a structured kernel-based method with a Markovian classifier, capitalizing the contribution of the dialogue modeling in deciding the speech act of individual sentences. One important finding of this evaluation is that a quite robust speech classifier can be obtained with almost no requirement in term of task-specific feature and system engineering: results are appealing mostly considering the reduced size of the dataset. Further work is needed to improve the overall F1 scores, possibly extending the adopted kernel function by addressing other dimensions of the linguistic information or also making the kernel more sensitive to task-specific knowledge. Also the combination of the adopted strategy with recurrent neural approaches is an interesting research direction.

References

Y. Altun, I. Tsochantaridis, and T. Hofmann. 2003. Hidden Markov support vector machines. In *Proceedings of the International Conference on Machine Learning*.

Michael Collins and Nigel Duffy. 2001. Convolution kernels for natural language. In *Proceedings of Neural Information Processing Systems (NIPS'2001)*, pages 625–632.

Danilo Croce, Alessandro Moschitti, and Roberto Basili. 2011. Structured lexical similarity via convolution kernels on dependency trees. In *Proceedings of EMNLP*.

Danilo Croce, Alessandro Moschitti, Roberto Basili, and Martha Palmer. 2012. Verb classification using distributional similarity in syntactic and semantic structures. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 263–272.

Simone Filice, Giuseppe Castellucci, Danilo Croce, and Roberto Basili. 2015. Kelp: a kernel-based learning platform for natural language processing. In *Proceedings of ACL-IJCNLP 2015 System Demonstrations*, pages 19–24.

Simone Filice, Giuseppe Castellucci, Giovanni Da San Martino, Alessandro Moschitti, Danilo Croce, and Roberto Basili. 2018. Kelp: a kernel-based learning platform. *Journal of Machine Learning Research*, 18(191):1–5.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Alessandro Moschitti. 2006. Efficient convolution kernels for dependency and constituent syntactic trees. In *ECML*, Berlin, Germany.

Nicole Novielli and Pierpaolo Basile. 2018. Overview of the evalita 2018 italian speech act labeling (ilisten) task. In Tommaso Caselli, Nicole Novielli, Viviana Patti, and Paolo Rosso, editors, *Proceedings of the 6th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA'18)*, Turin, Italy. CEUR.org.

Klaus Robert Müller, Sebastian Mika, Gunnar Rätsch, Koji Tsuda, and Bernhard Schölkopf. 2001. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2):181–201.

Magnus Sahlgren. 2006. *The Word-Space Model*. Ph.D. thesis, Stockholm University.

John Shawe-Taylor and Nello Cristianini. 2004. *Kernel Methods for Pattern Analysis*. Cambridge University Press.

David R. Traum, 1999. *Speech Acts for Dialogue Agents*, pages 169–201. Springer Netherlands, Dordrecht.

Vladimir N. Vapnik. 1998. *Statistical Learning Theory*. Wiley-Interscience.

S.V.N. Vishwanathan and Alexander J. Smola. 2002. Fast kernels on strings and trees. In *Proceedings of Neural Information Processing Systems*, pages 569–576.