This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2017.2657690, IEEE Transactions on Computational Intelligence and AI in Games

Learning to Extract Action Descriptions from Narrative Text

Oswaldo Ludwig, Quynh Ngoc Thi Do, Cameron Smith, Marc Cavazza, Marie-Francine Moens

***Abstract*—This paper focuses on the mapping of natural language sentences in written stories to a structured knowledge representation. This process yields an exponential explosion of instance combinations since each sentence may contain a set of ambiguous terms, each one giving place to a set of instance candidates. The selection of the best combination of instances is a structured classification problem that yields a high-demanding combinatorial optimization problem which, in this paper, is approached by a novel and efficient formulation of a genetic algorithm, which is able to exploit the conditional independence among variables, while improving the parallel scalability. The automatic rating of the resulting set of instance combinations, i.e. possible text interpretations, demands an exhaustive exploitation of the state-of-the-art resources in natural language processing to feed the system with pieces of evidence to be fused by the proposed framework. In this sense, a mapping framework able to reason with uncertainty, to integrate supervision, and evidence from external sources, was adopted. To improve the generalization capacity while learning from a limited amount of annotated data, a new constrained learning algorithm for Bayesian networks is introduced. This algorithm bounds the search space through a set of constraints which encode information on mutually exclusive values. The mapping of natural language utterances to a structured knowledge representation is important in the context of game construction, e.g. in an RPG setting, as it alleviates the manual knowledge acquisition bottleneck. The effectiveness of the proposed algorithm is evaluated on a set of three stories, yielding nine experiments. Our mapping framework yields performance gains in predicting the most likely structured representations of sentences when compared with a baseline algorithm.**

***Index Terms*—Intelligent narrative, natural language processing, structured prediction, constrained learning.**

I. Introduction

THE narrative provides a model for communicating experience and culture. Automatically extracting structured information from narrative text is a challenging task, since the structured representation of connected events and behaviors may involve commonsense inferences based on background knowledge, such as the semantic representation of objects, their properties and behavior, the motivations and goals behind the actions of characters, their emotional outcomes, and the actions they can undertake in the environment.

O. Ludwig, Q. T. N. Do and M. F. Moens are with the Department of Computer Science, Katholieke Universiteit Leuven, Belgium (e-mail: oswaldoludwig@gmail.com; quynhngocthi.do@cs.kuleuven.be; sien.moens@cs.kuleuven.be).

C. Smith is with the School of Computing, Teesside University, UK.

M. Cavazza is with the School of Engineering and Digital Arts, University of Kent, UK, (e-mail: M.O.Cavazza@kent.ac.uk).

The main research question of this paper is whether it is possible to provide a specific structured representation for narratives by fusing information from different sources and bounding the domain to a finite set of actions within the context of the current narrative. In this sense, this paper reports the results of our work on the knowledge representation for virtual worlds to answer the question "Who did What to Whom, and How, When and Where?", similar to the current Semantic Role Labeling (SRL) algorithms [1]. However, the SRL aims at a general-purpose semantic representation, i.e. it aims at providing a semantic representation at a higher-level of abstraction, while our work aims at instantiating semantic frame elements at a lower-level of abstraction, in an annotation style tailored for the narrative text. Therefore, we model the problem as a structured prediction task within a framework able to incorporate other sources of information, besides the text and the language model, to deal with the challenging task of instantiating semantic frame elements at lower-level of abstraction. The statistical reasoning is carried out by a special formulation of a genetic algorithm, which exploits the conditional independence between variables.

The paper is organized as follows. Section II presents the state of the art; Section III contextualizes the instantiation problem resulting from the translation. To ease the understanding of the proposed framework, Section IV introduces the basic ideas and a high-level diagram of the proposed approach with its different constituent parts. Once the context and details of the proposed framework have been explained, Section V provides the motivation for our approach. The adopted statistical model and features are described in Sections VI and VII respectively, while the statistical reasoning and the learning method are described in Sections VIII and IX respectively. The NLP pipeline is evaluated in Section X. Finally, Section XI presents the conclusions.

II. State of the art

There have been efforts in information extraction from textual sources, where the goal is to identify specific semantic components, such as people, objects, and actions, whose types are known ahead of time. Typically in information extraction [2] semantic labels are defined beforehand, and data are collected to train machine learning classifiers. On the sentence level, there are several schemes for recognizing the basic semantic roles of the sentence constituents, i.e. the who, does what, where, when constituents, the

most popular approaches being based on PropBank [3] and FrameNet [4] labels and their associated annotated corpora [5]. This entails work on finding the arguments of a semantic frame that is verb-centered, i.e. where the action or state is expressed by a verb in the sentence, and noun-centered. Some works, such as [6], aim at determining the character intentions, to provide the motivations for the actions performed. In the first instance this information can be useful in supporting the narrative interpretation, but in a second instance it can also improve the accuracy in predicting the correct action [7]. Our current framework does not model the character intentions; however, it makes possible to model these intentions, besides complex temporal, spatial or causal relationships in its Bayesian network based modeling.

On the discourse level, two recent tasks are the identification of events and entities that have a temporal or spatial impact and the linking of such events and entities with temporal or spatial relations. Researchers have been interested in building such models for decades [8], but recent progress has been encouraged by the construction of corpora like the TimeBank, [9], and corpora with spatial information [10], which provide events and times beyond temporal and spatial relations, annotated on English data. Researchers have also investigated methods for modeling sequences of events using recurrent neural networks [11].

Another important task is assigning narrative roles to characters in stories, since it can help in improving the accuracy of the structured representation of the narrative, e.g. by modeling the relationship between the characters through graphical models encoding latent variables representing the character role in the narrative. In [12] the authors propose to combine NLP techniques with narrative domain knowledge in order to automatically identify characters and their roles in the story according to Propp's theory [13], in which the character role is categorized into broad categories, such as hero, villain, dispatcher, donor, magical helper, prize, and false hero. In this sense, it is also important to identify mental affect states. The work [14] introduced the plot units as a structured knowledge representation for narrative stories. Plot units focus on the affect states of characters and the tensions between them. To automatically produce plot unit representations for narrative text, some works use affect projection rules to map the affect states onto the characters in the story [15]. To do so, they create a lexicon consisting of patient polarity verbs (PPVs) that reflect world knowledge about desirable/undesirable states for animate beings. A large corpus of narratives deeply-annotated according to Vladimir Propp's theory was made available as a result of the work of Finlayson [16].

The machine learning method adopted in this work, i.e. Bayesian network, has yielded reliable results in modeling narrative reasoning. For instance, in [17] the authors introduce a framework for machine-learning director agent strategies from observations of human-to-human interactions in an educational interactive narrative. The work utilized a Wizard-of-Oz paradigm where human wizards directed participants through Crystal Island's mystery storyline by dynamically controlling narrative events in the game. Interaction logs yielded training data to model the conditional probabilities of a dynamic Bayesian network model of the human wizards' directorial actions, achieving higher performance than naive Bayes and bi-gram model techniques.

Text understanding also involves coreference resolution, i.e. to identify when two mentions of an entity refer to the same thing or person in the real world [18], for instance, recognizing the entity to which him and it refer in the discourse, which is context-dependent, so many different interpretations of a text are possible.

III. Overview on the Instantiation

The mapping framework focuses on the problem of low-level concept instantiation, as required by the game engine that generates the animations. The low-level concept instantiation yields an exponential explosion of instance combinations, usually related to a large uncertainty, demanding a large amount of information to select the optimal combination. Therefore, the mapping framework bounds the search space according to the story context, in order to decrease the number of feasible combinations, and so the required amount of information.

In order to illustrate the problem, let us consider the sentence, "Tuk helped his father take care of his hunting tools", from the story "The Day Tuk Became a Hunter" [19], which is placed in an Eskimo community. By applying a current SRL algorithm, it is possible to recognize the events, to help and to take care, and their participants, Tuk and his father; however, to take care is a high-level representation of action; it must be instantiated by a low-level representation before providing it to the game engine that generates the animation.

Assuming that the hunting tools of the Eskimos have blades, the action/predicate "to take care" could be instantiated as to sharpen, which combined with the action "to help" brings to mind a scene in which Tuk and his father, named Nanuk, are sharpening the tools together. Therefore, an acceptable translation could be the set of concurrent events, $S = \{$SharpenItem(Tuk, knife), SharpenItem(Nanuk, knife)$\}$.

The above translation assumes that the system has information about the social network, father(Nanuk, Tuk), the relationship among objects, kindof(knife, hunting tool), and the relationship among actions, kindof(to sharpen, to take care). Part of that background information is entered directly into the system, such as the social network and the sets of characters, objects, locations, and actions belonging to the narrative. The kindof relationship among objects in different levels of knowledge representation (KR) are currently given by a language model based on neural

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2017.2657690, IEEE Transactions on Computational Intelligence and AI in Games

networks [20]; the same approach was adopted for actions in different levels of the KR. The relationship of pertinence between the actions and their arguments is encoded in a lookup table, in order to set to zero the conditional probability of the unfeasible arguments, saving processing time and improving the performance on unseen data.

Successfully comprehending stories involves gathering a much larger amount of background knowledge [21], which it is not within the scope of this paper. We are currently researching a semi-supervised cross-modal knowledge extraction method involving visual and textual sources [22]; however, there are other approaches, such as the one proposed in [23], where the authors describe a method for acquiring background knowledge through crowdsourcing, demonstrating how Games With A Purpose (GWAPs) can be used to acquire background knowledge.

Even assuming that the system has all the information required by the mapping process, the high computational cost, derived from the exponential explosion of combinations of entities and actions, is still a problem. For instance, let us assume that the system has only two instances of hunting tools and two instances of the action "to take care", more specifically, assuming that the system has the information: kindof(knife, hunting tool), kindof(spear, hunting tool), kindof(to sharpen, to take care), and kindof(to carry, to take care), the sentence in question would yield 20 feasible hypotheses, as shown in Table I.

TABLE I
DIFFERENT HYPOTHESES FOR A SENTENCE INTERPRETATION

#	sets of low-level actions and arguments
1	{SharpenItem(Tuk, knife), SharpenItem(Nanuk, knife)}
2	{SharpenItem(Tuk, knife), SharpenItem(Nanuk, spear)}
3	{SharpenItem(Tuk, spear), SharpenItem(Nanuk, knife)}
4	{SharpenItem(Tuk, spear), SharpenItem(Nanuk, spear)}
5	{CarryItem(Tuk, spear), SharpenItem(Nanuk, knife)}
6	{CarryItem(Tuk, spear), SharpenItem(Nanuk, spear)}
7	{CarryItem(Tuk, knife), SharpenItem(Nanuk, knife)}
8	{CarryItem(Tuk, knife), SharpenItem(Nanuk, spear)}
9	{SharpenItem(Tuk, knife), CarryItem(Nanuk, knife)}
10	{SharpenItem(Tuk, knife), CarryItem(Nanuk, spear)}
11	{SharpenItem(Tuk, spear), CarryItem(Nanuk, knife)}
12	{SharpenItem(Tuk, spear), CarryItem(Nanuk, spear)}
13	{CarryItem(Tuk, knife), CarryItem(Nanuk, knife)}
14	{CarryItem(Tuk, knife), CarryItem(Nanuk, spear)}
15	{CarryItem(Tuk, spear), CarryItem(Nanuk, knife)}
16	{CarryItem(Tuk, spear), CarryItem(Nanuk, spear)}
17	{SharpenItem(Tuk, knife)}
18	{SharpenItem(Tuk, spear)}
19	{CarryItem(Tuk, knife)}
20	{CarryItem(Tuk, spear)}

According to the context, one of those hypotheses is likely to be more in accordance with the reasoning of the author of the story than the others. Therefore, the algorithm proposed here adopts a joint probability function for the actions and their arguments, in order to select the optimal hypothesis. The adopted joint probability function accepts features about the social network, the relationship among the arguments, which can be summarized by the set $R = \{\text{A-Kind-Of, Is-A, Part-Of, Has-A}\}$, and the relationship among actions, which can be bounded to the subset $R_a = \{\text{A-Kind-Of, Is-A}\}$.

IV. THE TOOL PIPELINE

This section provides a top-down description of the tool pipeline and a functional description of the inputs and outputs of each stage.

As an overview of our tool, the system receives as input the narrative text in addition to the narrative domain, i.e. the sets of allowable slot values for the variables representing the action and its arguments, i.e. characters/avatars, items/objects, tools and movement directions, in accordance with the elements defined in the graphical framework. The output is a 3D animation of the provided text. The proposed framework starts by extracting a set of cues from the text by using state-of-the-art algorithms for natural language processing (NLP) (see the blocks Syntactic Processing, SRL and Coreference Resolution in Figure 1). This set of cues, henceforward represented by f, is composed by tokens corresponding to syntactic and semantic labels, such as subject, verb, and PropBank roles. This information is encoded in an XML file that is provided to the Mapping to KR module, which also receives the allowable variable values, i.e. the domain.

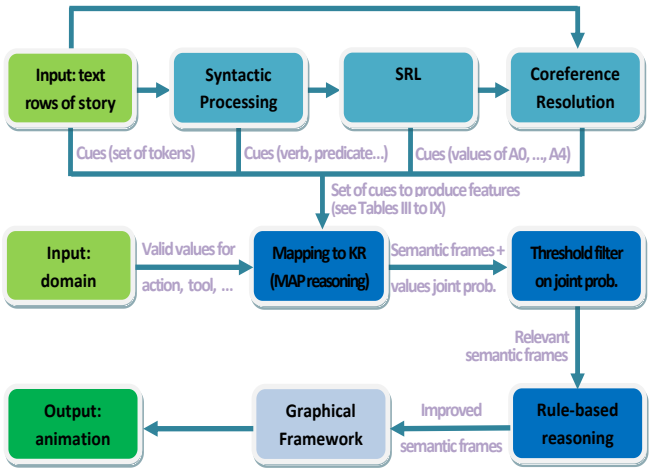

Fig. 1. The pipeline of the narrative processing, showing the NLP preprocessing, the mapping to KR and animation rendering.

The Mapping to KR module extracts vector representations from the set of cues, i.e. the labeled tokens, by using a recurrent neural network based language model [20]. Having such vector representations, features can be extracted from candidate values of each discrete variable, as explained in Section VII. The set of features is applied in modeling the probability distribution of the output variables, which are arranged in a Bayesian network (see Section VI), composing a structure, i.e. a semantic frame. After training (see Section IX), the Bayesian network is used as an objective function of a combinatorial optimization algorithm that chooses the best combination of allowable variable values, i.e. the best interpretation of the text, by maximizing the joint a posteriori probability provided by such a Bayesian model (see Section VIII). Therefore, the Mapping to KR module performs statistical reasoning and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2017.2657690, IEEE Transactions on Computational Intelligence and AI in Games

outputs a set of instantiated semantic frames, i.e. a structured KR of the narrative, as well as the respective values of the joint a posteriori probability, which are used by the next processing module to filter the irrelevant semantic frames by thresholding.

The produced semantic frames are post-processed by a simple rule-based inference engine that applies a set of deterministic rules encoding common sense information, such as "if X is listening Y, then Y is talking". Although this rule seems evident for humans, it is challenging for machines.

V. Motivation for our Approach

The instantiation problem posed here is a structured classification problem, which can be carried out by two main groups of algorithms, generative and discriminative. Among the generative methods we highlight a probabilistic graphical model, i.e. Bayesian networks, while from the discriminative methods we can highlight structured SVM (SSVM) and Markov random fields (MRF) [24].

In the case of the generative approach, a joint probabilistic model over the set of variables is defined, learned, and inferred/instantiated by using a probabilistic inference algorithm, in our case a combinatorial optimization based on GA, in order to classify the most likely joint assignment to all of the labels simultaneously, exploiting the correlations between them and encoding prior knowledge into the model. Regarding the MRF, it is important to highlight some advantages of this model over the Bayesian network, such as the possibility of representing cyclic dependencies; however, Bayesian networks can represent induced dependencies, which are more relevant for our application and cannot be modeled by MRF.

From the above options, a generative model was chosen, more specifically Bayesian Networks, since we have few annotated data, which demands the encoding of prior knowledge into the model, such as the causal relationship among variables. Regarding the training method, this work proposes a new training method for Bayesian networks that bounds the search space by using human knowledge, as detailed in Section IX, which is an alternative approach to the margin-maximizing properties of SSVM. However, the proposed training method is lighter than the SSVM training, which is highly computationally demanding in the context of our application, since each training example yields a combinatorial optimization problem on an exponentially large search space. Notice that even applying the cutting plane algorithm [25] to determine the most violated constraint, the SSVM training yields a hard combinatorial optimization problem per training sample per iteration.

Regarding the inference, in the case of Bayesian networks the decoding algorithm can exploit the sparsity of the model, i.e. it can exploit the conditional independence between variables, as will be shown in Section VIII. The same is not possible for SSVM, in which there is no sparsity to be exploited, yielding a higher computational cost. Moreover, the combinatorial optimization by GA, proposed here, plays an important role in giving parallel scalability to the inference system of any structured classification algorithm. Note that the most computational demanding task is the calculation of the fitness value of the GA individuals, which can be carried out independently of each other, enabling the parallelization of the code by sharing tasks even among hundreds of processors.

VI. The Statistical Model

From the machine learning point of view, the proposed mapping is the structured output prediction problem of learning a function

$$h : \mathcal{F} \rightarrow \mathcal{X} \tag{1}$$

where $\mathcal{F}$ is the space of inputs, in our case the set of cues, f, extracted from the text through state-of-the-art NLP algorithms, and $\mathcal{X}$ is a space of multivariate and structured outputs, whose elements are semantic frames of which the arguments depend on the predicate/action, which in turn depends on the predicate of the previous frame, to improve the consistency in the course of actions predicted by the mapping. Figure 2 illustrates the adopted graphical model.

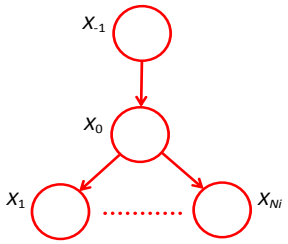

Fig. 2. The graphical model of the adopted mapping framework, in which X_{-1} represents the action of the previous semantic frame, X_0 the action of the current frame, and $X_1 \dots X_{N_i}$ their arguments.

The proposed mapping framework is based on log-linear models of the conditional probabilities of the actions/predicates and their arguments, where the variable X_{-1} represents the action of the previous semantic frame, X_0 the action of the current frame, and $X_1 \dots X_{N_i}$ their arguments, as can be seen in Figure 2. The variable values are represented here as $x_{(q,i)}$, where $q = -1 \dots N_i$ is the index of the variable and i is the index of its discrete value into the finite set of values S_q, $(q = -1 \dots N_i)$. In this sense, the conditional probability of the i^{th} discrete value, $x_{(q,i)}$, of the q^{th} variable, $X_q \in S_q$, given the state of its parents, Pa_q, the set of cues, f, and the adjustable parameter vector, θ_q, is modeled as follows:

$$P\left(X_q = x_{(q,i)} \mid Pa_q, f; \theta_q\right) = \frac{e^{\theta_q \phi_q\left(x_{(q,i)}, Pa_q, f\right)}}{\sum_{h=1}^{|S_q|} e^{\theta_q \phi_q\left(x_{(q,h)}, Pa_q, f\right)}} \tag{2}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2017.2657690, IEEE Transactions on Computational Intelligence and AI in Games

where $\phi_q(\cdot)$ is a m-dimensional feature function that extracts features from the set of cues, f, given the state of the variable X_q and the state of its parents, Pa_q, as detailed in Section VII. Alternatively, $\phi_q(\cdot)$ can be thought of as a compatibility function that measures how compatible the set of cues, f, the variable value, $x_{(q,i)}$, and the set of discrete values of its parents, Pa_q, are. From a linear algebra point of view, $\phi_q(\cdot)$ can also be understood as a basis function parametrized by θ_q.

The variables are related by a Bayesian network; therefore, it is possible to calculate the joint probability of the variables, given the set of cues, f, and the set of adjustable parameters, as follows:

$$P(X_{-1}\dots X_{N_i} \mid f;\theta_{-1}\dots\theta_{N_i}) = \prod_{q=-1}^{N_i} P(X_q \mid Pa_q, f;\theta_{-1}\dots\theta_{N_i}) \quad (3)$$

VII. Feature Extraction

The mapping algorithm extracts features by using cues provided by algorithms for SRL, part-of-speech tagging (POS), coreference resolution and a lookup table encoding word representations, extracted from a previously trained recurrent neural network based language model [20]. The lookup table, henceforth represented by $g: D \mapsto W$, maps each word w, defined in a dictionary D, which contains 82390 words, to an 80-dimensional space $W \subset R^{80}$, where similar words tend to be close to each other, making it possible to perform analogical reasoning using simple vector arithmetic [20]. For example, it is possible to answer a question, such as "What is the word that is similar to $small$ in the same sense as $biggest$ is similar to big?", by computing the vector $w = g(biggest) - g(big) + g(small)$, and searching for the word with the smallest cosine distance to w, which, if the model was well trained, is $smallest$. In this section, we describe the SRL and coreference resolution tools used in this work.

A. Semantic Role Labeling

We use the semantic parser from Lund university to detect semantic frames and their semantic roles under the PropBank formalism [26]. The algorithm assigns roles (from which the most frequent are A0-4, as can be seen in Table II) to sets of words, which are understood as arguments of the verb, assumed as the predicate; therefore, PropBank is a verb-oriented resource. That formalism also adopts modifier tags such as AM-LOC, about the location, and AM-DIR, about the direction, which are relevant for our mapping algorithm.

TABLE II
Most likely meaning of the PropBank semantic roles

role	description
A0	agent
A1	patient, theme
A2	indirect object, beneficiary, instrument, attribute, end state
A3	start point, beneficiary, instrument, attribute
A4	end point

B. Coreference Resolution

We use the coreference resolution tool from the LIIR lab of KU Leuven [18] to extract the links between the characters and the pronouns in the text. That tool detects entities and entity mentions, creating links between them. For example, given the text "This is the story of a boy named Tuk who lived in the Arctic. He wanted to show that he could be brave by hunting for big animals like his father who was a great hunter", the mentions "boy", "tuk" and the three pronoun mentions (He, he, his) are clustered as one entity, and the mentions "father" and "hunter" as the second entity.

C. Features of the current action (X_0)

Let S be the set of the words belonging to the sentence, `A0-4` $\subset D$, `AM-LOC` $\subset D$, and `AM-DIR` $\subset D$ be sets of words representing the respective SRL roles, S_1 be the set of low-level instances of characters, S_3-S_5 be the sets of low-level instances objects/items, tools, and locations, respectively, all of which are bounded by the story context. A 5-dimensional feature function, $\phi\left(f, x_{(0,i)}\right)$, is applied to model the probability of the current action. That feature function receives the instances of the current and previous actions and the set of cues, f, which is composed by the words of the sentence, the set `A1` $\in D$, and the verbs given by the PropBank-SRL. The action features are calculated as summarized in Table III, where $transitive\left(x_{(0,i)}\right) = 1$ if $x_{(0,i)}$ is a transitive verb; otherwise $transitive\left(x_{(0,i)}\right) = 0$. The logical operator $and(\cdot,\cdot) = 1$, if both arguments are true, otherwise $and(\cdot,\cdot) = 0$, and

$$z(a,b) = \frac{a^T b}{\|a\|\,\|b\|} \quad (4)$$

is the cosine similarity between two vectors, a and b.

TABLE III
Description of the features of the current action (X_0)

#	Description of the elements of $\phi\left(f, x_{(0,i)}\right)$
ϕ_1	$z\left(g\left(x_{(0,i)}\right), g(v)\right)$, where v is the verb given by SRL;
ϕ_2	$z\left(g\left(x_{(-1,j)}\right), g\left(x_{(0,i)}\right)\right)$, where $x_{(-1,j)}$ is the previous action;
ϕ_3	$and\left(transitive\left(x_{(0,i)}\right), \texttt{A1} \neq \varnothing\right)$, where `A1` is given by SRL;
ϕ_4	$max_j\left(z\left(g\left(x_{(0,i)}\right), g(w_j)\right)\right)$, where w_j is the j^{th} non-verb word;
ϕ_5	$z\left(g\left(x_{(0,i)}\right), g(v_1) + g(v_2)\right)$, where v_1 and v_2 are successive verbs

The first feature is the cosine similarity between the low-level instance of action, $x_{(0,i)}$, and the verb detected by the SRL, while the second feature is the cosine similarity between $x_{(0,i)}$ and the previous action, $x_{(-1,j)}$, to give consistency in the course of actions predicted by the mapping. The third feature returns the consistency between the SRL labeling and the instance candidate, $x_{(0,i)}$. More specifically, if the low-level action $x_{(0,i)}$ is transitive, the SRL must detect an $A1$ role. The fourth feature is the consistency between the instance candidate, $x_{(0,i)}$, and the context; more specifically, it is the similarity between the instance, $x_{(0,i)}$, and its most similar non-verb word in the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2017.2657690, IEEE Transactions on Computational Intelligence and AI in Games

sentence, in the cosine sense. The last feature was included to aid the algorithm in dealing with semantic frames in which the verb phrase is presented as such, "Tuk **takes care of his hunting tools**". In this case, the algorithm adds the vector representations of the words "takes" and "care", in order to get the vector representation of "takes care", and compares the resulting vector with $g\left(x_{(0,i)}\right)$ [20].

Let $p : W \mapsto \Gamma \subset R^2$ be a function that outputs the two-dimensional principal component analysis (PCA) projections, i.e. only the two components with largest variance, for the set of word representations, W, given by the language model based on neural networks, $g : D \mapsto W$, composing the mapping $p \circ g : D \mapsto \Gamma$. Therefore, to illustrate the idea behind the last action feature, Fig. 3 demonstrates the two-dimensional PCA representation, in Γ, of the words "take" and "care", beyond some verbs in a low-level KR, and the vector composition $p \circ g\left(\texttt{take care}\right) = p \circ g\left(\texttt{take}\right) + p \circ g\left(\texttt{care}\right)$. In an Euclidean sense, the nearest low-level instances for "take care" are "sharp" and "carry", while the chosen instance would be "sharp", since it has the largest cosine similarity in relation to "take care".

Fig. 3. Two-dimensional PCA projection for the vector representation of the words "take" and "care", beyond some low-level action instances, in red, and the vector composition $p \circ g\left(\texttt{take care}\right) = p \circ g\left(\texttt{take}\right) + p \circ g\left(\texttt{care}\right)$ in blue.

D. Features of the previous action (X_{-1})

As can be seen in Table IV, the features of the model of the previous action are the same as the features of the current action, except for the feature ϕ_2 of Table III.

TABLE IV
DESCRIPTION OF THE FEATURES OF THE PREVIOUS ACTION (X_{-1})

#	Description of the elements of $\phi\left(f, x_{(-1,i)}\right)$
ϕ_1	$z\left(g\left(x_{(-1,i)}\right), g\left(v\right)\right)$, where v is the action given by SRL;
ϕ_2	$and\left(transitive\left(x_{(-1,i)}\right), \texttt{A1} \neq \oslash\right)$, where A1 is given by SRL;
ϕ_3	$max_j\left(z\left(g\left(x_{(-1,i)}\right), g\left(w_j\right)\right)\right)$, where w_j is the j^{th} non-verb word;
ϕ_4	$z\left(g\left(x_{(-1,i)}\right), g\left(v_1\right) + g\left(v_2\right)\right)$, where v_1 and v_2 are successive verbs

E. Argument features

This work adopts a KR scheme in which each action can have the following arguments: character subject, $x_{(1,l)} \in S_1$, character object, $x_{(2,l)} \in S_1$, object/item, $x_{(3,l)} \in S_3$, tool, $x_{(4,l)} \in S_4$, and direction, $x_{(5,l)} \in S_5$.

1) Character-subject features: Let T be a set containing all the tenses and persons of the verb to be, $eq\left(\cdot,\cdot\right) : D \times D \mapsto \{0,1\}$ be a binary function which returns 1 if its arguments are equal to each other, and 0 otherwise, and $pos\left(\cdot\right) : S_0 \mapsto N_+$ be a function which returns the position, in the sentence, of the verb which was instantiated by the current low-level action, $x_{(0,i)}$. Therefore, the character-subject features, can be summarized as found in Table V.

TABLE V
DESCRIPTION OF THE CHARACTER-SUBJECT FEATURES

#	Description of the elements of $\acute{\phi}\left(x_{(0,i)}, f, x_{(1,l)}\right)$
$\acute{\phi}_1$	$\sum_{n=1}^{\lvert\texttt{A0}\rvert} eq\left(x_{(1,l)}, w_n\right)$, where $w_n \in$ A0;
$\acute{\phi}_2$	$-\sum_{n=1}^{\lvert\texttt{A1}\cup\texttt{A2}\rvert} eq\left(x_{(1,l)}, w_n\right)$, where $w_n \in (\texttt{A1} \cup \texttt{A2})$;
$\acute{\phi}_3$	$\sum_{n=1}^{\lvert S\rvert} eq\left(x_{(1,l)}, w_n\right)$, where $w_n \in S$ is the n^{th} word;
$\acute{\phi}_4$	$n - k$, where $k = pos\left(x_{(0,i)}\right)$ and n is the position of $x_{(1,l)}$;
$\acute{\phi}_5$	$\sum_{n=1}^{\lvert S\rvert}\sum_{j=1}^{\lvert T\rvert} eq\left(w_n, t_j\right)$, where $w_n \in S$, $t_j \in T$;
$\acute{\phi}_6$	$\left(1 - 2\acute{\phi}_5\right)\acute{\phi}_4$, a cross-term.

The first feature returns a value larger than zero if the current instance, index l, belongs to the set A0, given by the SRL, i.e. it gives information about whether the instance is likely to be an agent. The second feature is about the consistency between the SRL labeling and the instance, $x_{(1,l)}$. This feature is particularly important when the SRL fails in detecting the A0, i.e. when $\texttt{A0} = \oslash$. Since the action demands a subject, the second feature was included to distinguish characters present in the sentence which aren't present in A1 or A2, which are likely to be the character subject. The third feature returns how many times the instance, $x_{(1,l)}$, is present in the sentence, thus if its output is zero, then $x_{(1,l)}$ isn't the correct instance. The fourth feature is about the position of the instance, $x_{(1,l)}$, in relation to the verb in question, i.e. the verb which corresponds to the current low-level action instance, $x_{(0,i)}$. The fifth feature is a cue of the passive voice usage, a situation in which the relative positions of the subject and the verb may be inverted. The last feature is a cross-term between the fourth and fifth features, the idea is to invert the sign of the distance, $\acute{\phi}_4$, between $x_{(1,l)}$ and the verb in the case of passive voice usage. The cross-term is required because the adopted log-linear model cannot compute such non-linear composition.

2) Character-object features: Table VI summarizes the features extracted in modeling the probability function of the character-object, which are similar to the ones extracted for the character-subject, except for the inversion of the position of the first two features and the inversion of the sign of the features $\acute{\phi}_1$, $\acute{\phi}_2$, and $\acute{\phi}_4$, which are adopted only for the sake of clarity, since those changes make no difference for the training algorithm.

3) Item/object features: As can be seen in Table VII, the first item/object feature is about the labeling of $x_{(3,l)}$ as belonging to A2 by the SRL, i.e. whether $x_{(3,l)}$ was labeled

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2017.2657690, IEEE Transactions on Computational Intelligence and AI in Games

TABLE VI
DESCRIPTION OF THE CHARACTER-OBJECT FEATURES

#	Description of the elements of $\acute{\phi}\left(x_{(0,i)}, f, x_{(2,l)}\right)$
$\acute{\phi_1}$	$\sum_{n=1}^{\lvert \texttt{A1} \cup \texttt{A2} \rvert} eq\left(x_{(2,l)}, w_n\right)$, where $w_n \in (\texttt{A1} \cup \texttt{A2})$;
$\acute{\phi_2}$	$-\sum_{n=1}^{\lvert \texttt{A0} \rvert} eq\left(x_{(2,l)}, w_n\right)$, where $w_n \in \texttt{A0}$;
$\acute{\phi_3}$	$\sum_{n=1}^{\lvert S \rvert} eq\left(x_{(2,l)}, w_n\right)$, where $w_n \in S$ is the n^{th} word;
$\acute{\phi_4}$	$k-n$, where $k = pos\left(x_{(0,i)}\right)$ and n is the position of $x_{(2,l)}$;
$\acute{\phi_5}$	$\sum_{n=1}^{\lvert S \rvert} \sum_{j=1}^{\lvert T \rvert} eq\left(w_n, t_j\right)$, where $w_n \in S$, $t_j \in T$;
$\acute{\phi_6}$	$\left(1-2\acute{\phi_5}\right)\acute{\phi_4}$, a cross-term.

TABLE IX
DESCRIPTION OF THE DIRECTION FEATURES

#	Description of the elements of $\acute{\phi}\left(x_{(0,i)}, f, x_{(5,l)}\right)$
$\acute{\phi_1}$	$\sum_{n=1}^{\lvert \texttt{A4} \rvert} eq\left(x_{(5,l)}, w_n\right)$, where $w_n \in \texttt{A4}$;
$\acute{\phi_2}$	$\sum_{n=1}^{\lvert \texttt{AM-DIR} \rvert} eq\left(x_{(5,l)}, w_n\right)$, where $w_n \in \texttt{AM-DIR}$;
$\acute{\phi_3}$	$-\sum_{n=1}^{\lvert \texttt{A2} \cup \texttt{A3} \rvert} eq\left(x_{(5,l)}, w_n\right)$, where $w_n \in (\texttt{A2} \cup \texttt{A3})$;
$\acute{\phi_4}$	$\sum_{n=1}^{\lvert S \rvert} eq\left(x_{(5,l)}, w_n\right)$, where $w_n \in S$ is the n^{th} word;
$\acute{\phi_5}$	$z\left(g\left(x_{(0,i)}\right), g\left(x_{(5,l)}\right)\right)$, where $x_{(0,i)}$ is the current action;
$\acute{\phi_6}$	$k-n$, where $k = pos\left(x_{(0,i)}\right)$ and n is the position of $x_{(5,l)}$.

as an indirect object or beneficiary. The second feature is about the consistency between the SRL labeling and the assumption of $x_{(3,l)}$. The third feature is about the presence of the current instance in the sentence. The fourth feature is about the consistency between the instance candidate, $x_{(3,l)}$, and the context; more specifically, it is the similarity between the action, $x_{(0,i)}$, and the current instance, in the cosine sense. The last feature is about the position of the instance, $x_{(3,l)}$, in relation to the verb.

TABLE VII
DESCRIPTION OF THE ITEM/OBJECT FEATURES

#	Description of the elements of $\acute{\phi}\left(x_{(0,i)}, f, x_{(3,l)}\right)$
$\acute{\phi_1}$	$\sum_{n=1}^{\lvert \texttt{A2} \rvert} eq\left(x_{(3,l)}, w_n\right)$, where $w_n \in \texttt{A2}$;
$\acute{\phi_2}$	$-\sum_{n=1}^{\lvert \texttt{A3} \rvert} eq\left(x_{(3,l)}, w_n\right)$, where $w_n \in \texttt{A3}$;
$\acute{\phi_3}$	$\sum_{n=1}^{\lvert S \rvert} eq\left(x_{(3,l)}, w_n\right)$, where $w_n \in S$ is the n^{th} sentence word;
$\acute{\phi_4}$	$z\left(g\left(x_{(0,i)}\right), g\left(x_{(3,l)}\right)\right)$, where $x_{(0,i)}$ is the current action;
$\acute{\phi_5}$	$k-n$, where $k = pos\left(x_{(0,i)}\right)$ and n is the position of $x_{(3,l)}$.

4) Tool features: As summarized in Table VIII, the features for the tool model are the same as the item/object features; however, the domain set is S_4 despite S_3.

TABLE VIII
DESCRIPTION OF THE TOOL FEATURES

#	Description of the elements of $\acute{\phi}\left(x_{(0,i)}, f, x_{(4,l)}\right)$
$\acute{\phi_1}$	$\sum_{n=1}^{\lvert \texttt{A2} \rvert} eq\left(x_{(4,l)}, w_n\right)$, where $w_n \in \texttt{A2}$;
$\acute{\phi_2}$	$-\sum_{n=1}^{\lvert \texttt{A3} \rvert} eq\left(x_{(4,l)}, w_n\right)$, where $w_n \in \texttt{A3}$;
$\acute{\phi_3}$	$\sum_{n=1}^{\lvert S \rvert} eq\left(x_{(4,l)}, w_n\right)$, where $w_n \in S$ is the n^{th} word;
$\acute{\phi_4}$	$z\left(g\left(x_{(0,i)}\right), g\left(x_{(4,l)}\right)\right)$, where $x_{(0,i)}$ is the current action;
$\acute{\phi_5}$	$k-n$, where $k = pos\left(x_{(0,i)}\right)$ and n is the position of $x_{(4,l)}$.

5) Direction features: The features adopted for the direction are similar to the location features; the first two features are about the SRL labeling, i.e. whether the instance is likely to be an end point or a direction (AM-DIR), while the third feature is useful when the SRL outputs a false negative of A4, i.e. when the information about the destination is present in the text, but the SRL returns A4 $= \oslash$; therefore, if the instance isn't a start point or an indirect object, it is likely to be a direction. The three last features are the same as those of the tool, as can be seen in Table IX.

VIII. MAXIMUM A POSTERIORI INFERENCE

Given the graphical model, the problem of low-level concept instantiation can be understood as the task of finding the most likely configuration of its variables, known as the maximum a posteriori (MAP) problem. In the present work, this NP-hard problem [27] is formulated as a combinatorial optimization problem whose objective function is in mathematical form, as follows:

$$\underset{X_{-1}, X_0, \ldots, X_{N_i}}{\arg\max} \; P\left(X_{-1}, X_0, \ldots, X_{N_i} \mid f; \theta_{-1}, \ldots, \theta_{N_i}\right) \tag{5}$$

where $P\left(X_{-1}, X_0, \ldots, X_{N_i} \mid f; \theta_{-1}, \ldots, \theta_{N_i}\right)$ is given by (3). This problem demands the evaluation of a large amount of hypotheses; more specifically, taking into account the adopted log-linear distribution function, the computational time complexity of an exhaustive search would be $O\left(n_f \times n^m\right)$, where n_f is the number of features of the log-linear distribution, n is the average number of instances per variable, and m is the number of variables.

The MAP problem has been approached using many methods, see [28] and [29]. We have approached this problem by using a GA [30]. Similar to other algorithms for metaheuristic optimization (MHO), the GA does not provide certificates of optimality; however, in our case, a near optimal solution can be obtained in a short time, since the GA provides parallel scalability because the fitness of the GA individuals can be calculated independently of one another. Moreover, our special GA formulation exploits the sparsity of the Bayesian networks, i.e. the conditional independence between the variables.

Resuming our notation, as we defined Pa_q as the set of parents of the node q, modeled by its conditional distribution $P\left(X_q \mid Pa_q\right)$, similarly we define Ch_q as the set of children of X_q. Our GA formulation exploits a BN property that arises when $Ch_q = \emptyset$. In this case, the state of X_q doesn't affect the conditional distribution of the other nodes, and the optimization for the node q can be carried out independently from the others, excepting the nodes belonging to the set Pa_q, which affects the conditional distribution of node q. Therefore, the GA formulation can be adapted to exploit a smaller search space. To do so, we split our set of variables $\{X_0, \ldots, X_{N_i}\}$ into two subsets; $\Omega = \left\{\widetilde{X}_1, \ldots, \widetilde{X}_{\widetilde{M}}\right\}$, where $\widetilde{X}_i \in \widetilde{S}_i$, $i = 1, \ldots, \widetilde{M}$, are the variables whose sets of children are empty, and $\Psi = \left\{\bar{X}_1, \ldots, \bar{X}_{\bar{M}}\right\}$, where $\bar{X}_i \in \bar{S}_i$, $i = 1, \ldots, \bar{M}$, are the variables that have, at least, one child.

The combinatorial optimization by GA assumes as fitness function the a posteriori probability (3), given by a Bayesian network previously trained on annotated data. The GA has a chromosome vector of dimension $\bar{M}$, in which each gene encodes the index of a state candidate of one of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2017.2657690, IEEE Transactions on Computational Intelligence and AI in Games

the variables belonging to the set Ψ. The chromosomes of the initial population are loaded in a uniform distribution, where the feasible values of the i^{th} gene are natural numbers bounded into the interval $[1, |S_i|]$. The evaluation of the fitness of each GA individual carries out a sub-searching process to find the state of the set of variables belonging to Ω that maximizes the fitness function. This sub-searching process can be carried out for each variable individually, requiring less processing power. Details on our formulation for this combinatorial optimization problem can be found in Appendix A, see (16)-(19), which are solved by Algorithm 1.

During the loop over generations the fitness value, Φ_{ind}, of each individual, ind, is evaluated according to (17) and (18). Then, the individuals are ranked according to their fitness values and the crossover operator is applied to generate new individuals by randomly selecting the parents by their ranks, according to the random variable proposed in our previous work [31], in which it is possible to set the selective pressure p. In our algorithm, the usual crossover operation was modified in order to deal with combinatorial optimization problems, namely, each gene of a new individual is randomly taken from one of the parents. This combinatorial optimization algorithm was adapted from an algorithm for feature selection[1] developed for our previous work [32].

Algorithm 1 returns the variable values that yield the largest value of the fitness function, i.e. the most likely structured representation of the current semantic frame, bounded by the given domain.

This work also contributes with a pre-processing method that decreases the computational cost of the MAP estimation. Let us consider a variable, X_q, whose set of parents is empty, i.e. $Pa_q = \emptyset$, in the case of our model represented by the variable X_{-1} (see Fig. 2). Since this work adopts a threshold on the joint a posteriori probability for rejecting semantic frames that are unlikely to be represented by the adopted KR schema, it is possible to speed up the combinatorial optimization by reducing, in advance, the cardinality of the set of discrete values, $|S_q|$, of X_q by exploiting the following property:

$$P\left(x_{(q,i)}|f;\theta_q\right) \geq P\left(x_{(1,i)},\dots x_{(q,i)}\dots,x_{(N_i,k)} \mid f;\theta_1,\dots\theta_q\dots\theta_{N_i}\right) \quad (6)$$

Notice that, if $P\left(x_{(q,i)}|f;\theta_q\right)$ is smaller than the adopted threshold, the joint a posteriori probability, represented by the right hand side of (6), also is. Therefore, given the set of cues, f, it is possible to reject in advance all discrete values belonging to the set S_q that yields $P\left(x_{(q,i)}|f;\theta_q\right)$ smaller than the adopted threshold, thus saving processing time during the combinatorial optimization.

[1]The original Matlab code is available for download at Matlab Central, http://www.mathworks.com/matlabcentral/fileexchange/29553-feature-selector-based-on-genetic-algorithms-and-information-theory

Algorithm 1 Combinatorial optimization by GA

Input: p, S, D, W, f, Ω, Ψ, N_{pop}: the selective pressure, p, the sentence, S, the dictionary, D, and its respective word representations, W, SRL and syntactic features, f, the sets of discrete variable values Ω, and the number of GA individuals, N_{pop}, respectively.
Output: X^*, Φ^*: a vector with the indices of the optimal states of the variables and the optimal value of the fitness function (for frame filtering by thresholding) respectively.
Generate a set with N_{pop} chromosomes $\{\mathbf{Cr}\}$ of dimension $\bar{M}$ for the initial population, in which each gene encodes the index of a state candidate of one of the variables belonging to the set Ψ, randomly generated in a uniform distribution, where the feasible values of the i^{th} gene are natural numbers bounded into the interval $[1, |S_i|]$;
for $generation = 1 : max_{gener}$ **do**
  // Evaluating the population: //
  for $ind = 1 : N_{pop}$ **do**
    $[\bar{x}_1,\dots,\bar{x}_{\bar{M}}] \leftarrow Cr_{ind}$: load the variables $\bar{X}_q$, $q = 1,\dots,\bar{M}$, with the indices stored in the chromosome of the current individual;
    for $j = 1 : \widetilde{M}$ **do**
      Exhaustive search for $\widetilde{x}_j^*$ according to (19);
    end for
    Substitute $\widetilde{x}_1^*,\dots,\widetilde{x}_{\widetilde{M}}^*$ and $\bar{x}_1,\dots,\bar{x}_{\bar{M}}$ into (17) and (18) to have the current values of Π_1 and Π_2;
    $\Phi_{ind} \leftarrow \Pi_1 + \Pi_2$: storing the fitness of individual ind;
  end for
  Rank the individuals according to their fitness Φ_{ind};
  Store/update the genes of the best individual in $\mathbf{Cr}^*$ and the last values of $\widetilde{x}_1^*,\dots,\widetilde{x}_{\widetilde{M}}^*$ into the output vector X^*;
  Store/update the best fitness value Φ^*;
  // Performing the crossover: //
  for $k = 1 : N_{pop}$ **do**
    // Randomly selecting the indices of parents by using the asymmetric distribution proposed in [31]: //
    $\vartheta_j \leftarrow$ random number $\in [0,1]$ with uniform distribution, $j = 1,2$;
    $parent_j \leftarrow round\left((N_{pop}-1)\,\frac{e^{p\vartheta_j}-1}{e^p-1}+1\right)$, $j = 1,2$;
    // Assembling the chromosome $\mathbf{Cr}_k^{son}$: //
    for $m = 1 : \bar{M}$ **do**
      Randomly select a parent (i.e. between $parent_1$ and $parent_2$) to give the m^{th} gene for the k^{th} individual of the new generation:
      $\mathbf{Cr}_{(k,m)}^{son} \leftarrow \mathbf{Cr}_{(parent_{1or2},m)}$;
    end for
  end for
end for

IX. Model Training

This section introduces a new constrained learning algorithm for Bayesian networks that yields a convex optimization problem. This algorithm makes it possible to include human knowledge in the training, thus helping in dealing with the limited amount of annotated data.

One of the ideas behind the mapping is to fuse information within a constrained domain, by training the mapping on annotated data sets of small cardinalities, only to adapt the algorithm to a given context. Therefore, beyond having few features, and so a small number of related parameters to be adjusted, the constraining of the search space is a key issue in keeping the generalization capacity.

Despite the popularity of the maximum margin principle [33] and its problem-independent geometric constraints, the mapping framework bounds the search space through a set of constraints encoding information on mutually exclusive values, i.e. information about the unlikeliness

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2017.2657690, IEEE Transactions on Computational Intelligence and AI in Games

of some conjunctions of variable states, or set of states, which are defined by the expert knowledge of the user, such as animals cannot talk or use tools, generating several constraints resulting from the combination of all the animals belonging to the domain and actions that they can not perform. Therefore, the mapping framework makes available a friendly user interface to input information on mutually exclusive values, henceforth called exclusivity constraints. These constraints are modeled in a statistical manner, i.e. for an ordered pair of variables, (X_m, X_n) having the values $(x_{(m,i)}, x_{(n,j)})$ and subject to the exclusivity constraint, the following constraint is assumed for each training example, k:

$$P\left(x_{(m,i)} \mid pa_{(m,k)}, f_k; \theta_m\right) P\left(x_{(n,j)} \mid pa_{(n,k)}, f_k; \theta_n\right) \leq \xi \quad (7)$$

where $pa_{(m,k)}$ represents the state of the parents of X_m in the observation k, f_k is the set of cues extracted from the same observation and ξ is an upper bound on the probability of above conjunction that is set by the user. Notice that, this constraint assumes that the larger the likelihood of $x_{(m,i)}$, the smaller the likelihood of $x_{(n,j)}$. The user interface allows defining sub-sets of exclusivity constraints at once. To do so, the user only has to provide two sets of slot values, one for variable X_m and the other of the variable X_n. The system automatically generates the constraints by giving all the pairwise combinations of the values per training example, i.e. one constraint per pairwise combination per training example.

The mathematical formulation of our new constrained learning method is detailed in Appendix B.

X. Experiments

In this section the mapping algorithm is evaluated on three stories: "The Day Tuk Became a Hunter" [19], "The Bear and the Travelers"[2] and "The First Tears"[3], henceforth referred to as story#1, story#2 and story#3, respectively. The idea is to have nine experiments, by training and evaluating on different stories, in order to assess the generalization capacity of the mapping framework, besides its capacity in fitting the training data, i.e. by evaluating also in the same story in which the algorithm was trained, in order to have information about the bias error, enabling the analysis of the bias-variance tradeoff, avoiding data overfitting. Therefore, the procedure is guided by four steps: 1) to input the domain of the training story (the set of characters, objects, tools and directions); 2) to train the Bayesian network, i.e. to adjust the parameters of the log-linear distributions on the training story; 3) to change from the domain of the training story to that of the testing story (inputting the set of characters, objects, tools and directions belonging to the testing story); 4) to evaluate the algorithm on the testing story with the parameters of the log-linear distributions previously adjusted on the training story.

[2]http://fairytalesoftheworld.com/quick-reads/the-bear-and-the-travellers/
[3]http://americanfolklore.net/folklore/2010/09/the_first_tears.html

TABLE X
Example of the SRL and Coreference Resolution outputs

"He practiced using a spear and even knew how to cut up animals"
SRL output
Frame#1 pred:practiced; A0:He; A1:using a spear Frame#2 pred:using; A0:He; A1:a spear Frame#3 pred:knew; A0:He; A1:how to cut up animals; AM-ADV:even Frame#4 pred:cut; A1:different animals
Coref. output
Frame #1 A0:Tuk Frame #2 A0:Tuk

TABLE XI
Example of the mapping output

"He practiced using a spear and even knew how to cut up animals"					
semantic frame #1					
action	**char−subj**	**char−obj**	**obj/item**	**tool**	**direction**
to practice	tuk	none	none	spear	none
semantic frame #4					
action	**char−subj**	**char−obj**	**obj/item**	**tool**	**direction**
to cut	tuk	none	animals	knife	none

Regarding the domain, story#1, yields a set S_0 composed of 88 possible actions, i.e. actions processable by the graphical framework, while story#2 yields $|S_0| = 28$ and story#3 yields $|S_0| = 34$. The adopted evaluation metrics were precision, recall, and F1. Since we introduced a new annotation scheme directly related with the task of animation rendering, it was not possible to compare our work with existing works based on other annotation schemes [34]; however, this section reports comparative experiments with our special formulation of GA for MAP reasoning against two baseline algorithms; the usual GA and random-restart hill climbing.

To contextualize the experiments, this section starts by exemplifying the mapping output. According to our KR scheme, the mapping output is a set of low-level instances of actions/predicates and their respective instance-arguments per semantic frame. Let us consider the sentence "he practiced using a spear and even knew how to cut up animals". From the SRL and the coreference resolution for the pronouns, see Table X, the mapping module recognizes two semantic frames which are relevant, the first frame is ruled by the predicate "practiced" and the second by the predicate "cut". For each relevant semantic frame the system outputs the value of the predicate/action and the set of argument values, as can be seen in the output example of Table XI.

The information of Table XI is encoded in an XML file, according to the XSD schema of Listing 1.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2017.2657690, IEEE Transactions on Computational Intelligence and AI in Games

TABLE XII
PERFORMANCE INDICES OF THE SEMANTIC ROLE CLASSIFICATION.

Role	precision	recall	F1
A0	0.78	0.72	0.75
A1	0.71	0.77	0.74
A2	0.48	0.50	0.49
AM-LOC	0.47	0.41	0.44
AM-TMP	0.65	0.63	0.64
AM-MNR	0.56	0.50	0.53
AM-DIR	0.75	0.47	0.58

Listing 1. The XSD schema definition of the output of mapping to KR.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="mapping">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="sentence">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="action">
       <xs:complexType>
        <xs:element name="char-subj" type="xs:string"/>
        <xs:element name="char-obj" type="xs:string"/>
        <xs:element name="item" type="xs:string"/>
        <xs:element name="tool" type="xs:string"/>
        <xs:element name="direction" type="xs:string"/>
        <xs:element name="JointProb" type="xs:decimal"/>
       </xs:complexType>
      </xs:element>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

Notice that the mapping is able to infer some information which is not present in the text, such as the tool used to cut animals (see Table XI), due to the feature $\acute{\phi}_4$ of Table VIII that exploits the language model, encoded in the lookup table $g : D \rightarrow W$, when computing the cosine similarity between the action "to cut" and the tool "knife".

To better evaluate the mapping results we first evaluate the outcome of the NLP algorithms. The performance of our coreference tool [18] was assessed by using the measure defined in the CoNLL 2011 coreference task, which is the average of three widespread measures (MUC, B^3 and $CEAF_e$). The result of the application of our coreference tool on the three stories, was $MUC = 0.918$, $B^3 = 0.744$, $CEAF_e = 0.516$ and $Avg = 0.726$. The results per role of the semantic role classification is given in Table XII.

Regarding the mapping trained with exclusivity constraints, Table XIII reports the experimental results obtained by combining training and evaluation in different stories, while Table XIV summarizes the statistics on the F1 values reported in Table XIII for two situations; when the model is evaluated on the same story than the one on which it was trained, i.e. the ground truth, and when the model is evaluated in other stories. The results of Table XIV can be compared with the results obtained by using the mapping trained by the usual maximum likelihood method, summarized in Table XV.

As can be seen in Table XIV, even the prediction of the action/predicate yields mistakes, due to the issues in

TABLE XIII
PERFORMANCE MAPPING TRAINED WITH EXCLUSIVITY CONSTRAINTS.

frame elem.	perf. index	train \ test	story#1	story#2	story#3
action/predicate	precision	story#1	0.93	0.92	1.00
		story#2	0.78	1.00	0.93
		story#3	0.85	0.85	1.00
	recall	story#1	0.86	0.80	0.74
		story#2	0.72	0.87	0.68
		story#3	0.79	0.73	0.79
	F1	story#1	0.89	0.86	0.85
		story#2	0.75	0.93	0.79
		story#3	0.82	0.79	0.88
character subject	precision	story#1	0.96	0.85	0.93
		story#2	0.80	0.92	0.87
		story#3	0.84	0.85	1.00
	recall	story#1	0.86	0.73	0.68
		story#2	0.71	0.80	0.68
		story#3	0.75	0.73	0.74
	F1	story#1	0.91	0.79	0.79
		story#2	0.75	0.86	0.76
		story#3	0.79	0.79	0.85
character object	precision	story#1	0.83	0.67	1.00
		story#2	0.50	1.00	1.00
		story#3	0.67	0.67	1.00
	recall	story#1	0.71	0.67	0.50
		story#2	0.43	1.00	0.50
		story#3	0.57	0.50	0.50
	F1	story#1	0.77	0.67	0.67
		story#2	0.46	1.00	0.67
		story#3	0.62	0.57	0.67
item/object	precision	story#1	0.80	1.00	0.88
		story#2	0.60	1.00	0.63
		story#3	0.70	0.40	0.88
	recall	story#1	0.67	0.50	0.78
		story#2	0.46	0.67	0.56
		story#3	0.58	0.67	0.88
	F1	story#1	0.73	0.67	0.83
		story#2	0.52	0.80	0.59
		story#3	0.63	0.50	0.88
tool	precision	story#1	1.00	-	1.00
		story#2	0.44	-	0.50
		story#3	0.78	-	1.00
	recall	story#1	0.90	-	1.00
		story#2	0.40	-	0.33
		story#3	0.70	-	1.00
	F1	story#1	0.95	-	1.00
		story#2	0.42	-	0.40
		story#3	0.74	-	1.00
direction	precision	story#1	0.83	0.50	1.00
		story#2	0.50	1.00	0.50
		story#3	0.67	0.50	1.00
	recall	story#1	0.63	0.33	0.20
		story#2	0.38	0.67	0.20
		story#3	0.50	0.33	0.40
	F1	story#1	0.72	0.40	0.33
		story#2	0.43	0.80	0.29
		story#3	0.57	0.40	0.57

TABLE XIV
F1 STATISTICS OF MAPPING TRAINED WITH EXCLUSIVITY CONSTRAINTS (MEAN $\pm$ STANDARD DEVIATION).

Frame element	ground truth	train and test in different stories
action/predicate	0.90±0.03	0.81±0.04
character subject	0.87±0.03	0.78±0.02
character object	0.81±0.17	0.61±0.08
item/object	0.80±0.08	0.62±0.12
tool	0.98±0.04	0.64±0.29
direction	0.70±0.12	0.40±0.10

TABLE XV
F1 STATISTICS OF MAPPING TRAINED BY MAXIMUM LIKELIHOOD (MEAN $\pm$ STANDARD DEVIATION).

Frame element	ground truth	train and test in different stories
action/predicate	0.91±0.03	0.79±0.05
character subject	0.87±0.03	0.76±0.02
character object	0.83±0.15	0.57±0.09
item/object	0.81±0.08	0.59±0.12
tool	0.99±0.02	0.60±0.29
direction	0.70±0.13	0.39±0.09

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2017.2657690, IEEE Transactions on Computational Intelligence and AI in Games

associating the verb(s) in the sentence with the actions belonging to the set of actions S_0; moreover, predicting the best instance for comprehensive actions, such as "to take care", which can be instantiated as "to sharpen" or "to carry", is also a problem, as can be seen in the example of Table I. To deal with these issues, the mapping makes use of information from the context (see feature ϕ_4 of Table III) beyond information from the language model, as illustrated in Fig. 3. Moreover, there are action instances belonging to S_0, such as "to give", which can be easily represented in some occurrences, such as in the sentence "Tuk's father gave him a new knife", but is unrepresentable in the case of the sentence "Tuk's father gave him many hunting tips".

By taking into account the tight relationship between the role A0 and the character subject, see Table II, and comparing the F1 value of the role A0 in Table XII with the mean value of F1 for the character subject (see the second line of Table XIV), it is possible to realize that the mapping has produced a slight improvement in recognizing the character subject, even when the model is trained and evaluated in different stories. It might be due to the limited domain and the fusion of information from different sources, e.g. information from the language model, information encoded in the feature extraction (see Tables III-IX), information encoded in the Bayesian model and in the constrained training. However, it is not possible to compare the performance of the other SRL rules with the mapping performance, since the PropBank annotation style is less specific than the annotation style assumed for the mapping.

Regarding the processing time during the prediction stage, the exploitation of the properties (6) and (15) enables a quick mapping through GA. Moreover, our MAP algorithm seems consistent, in the sense that it presents a small standard deviation on the CPU time, as can be seen in Table XVI, which summarizes the mean and standard deviation values of the CPU time demanded to solve the MAP problem for the chosen stories, running on quad-core processor, by using our special formulation of GA, henceforward called SGA, and two baseline algorithms: the usual GA (without exploiting the properties given by (6) and (15)) and random-restart hill climbing (RRHC). In this experiment the number of GA individuals and the number of restarting loops (in the case of RRHC algorithm) were chosen aiming at overcoming local minima, in such a way that the choice of the MAP algorithm has no impact on the performance indices. However, the choosing of the algorithm for MAP reasoning can strongly affect the CPU time, which is the subject of our evaluation.

The advantage of SGA over the usual GA could be theoretically predicted by comparing (14) and (20). Regarding RRHC, the major drawback seems to be the lack of an efficient meta-heuristic. This issue implies a large standard deviation on the distribution of the CPU time in experiments with repeated measures. The stop criterion of our RRHC implementation is based on a tolerance value, i.e. a threshold on the number of change attempts per variable without resulting improvement on the objective function; therefore, the CPU time can vary. It was also observed that the hill climbing algorithm demands several attempts to find a variable value that improves the objective function when the algorithm approaches a local optimum.

TABLE XVI
CPU TIME IN SECONDS (MEAN ± STANDARD DEVIATION), AVERAGE NUMBER OF GA INDIVIDUALS, GA GENERATIONS, RRHC RESTARTING LOOPS AND CHANGE ATTEMPTS/VARIABLE/LOOP.

	SGA	GA	RRHC
CPU time	14.28±0.39	44.70±0.37	42.51±8.66
# GA individuals	40	100	-
# GA generations	14.83	19.50	-
# RRHC restarting loops	-	-	50
# change attempts/variable/loop	-	-	38.74

XI. CONCLUSION

In this work, we introduced a framework to map text from written stories to a specific low-level KR. This new framework is able to reason with uncertainty, to integrate training from annotated data and constraints encoding information on mutually exclusive values, beyond evidence from external sources, such as information from the language model [20]. Similar to other methods for structured prediction, the mapping aims at predicting the most likely structure by searching in the large search space derived from the exponential explosion of instance combinations, i.e. MAP inference. Therefore, an algorithm based on GA, able to exploit some properties of the Bayesian network, see (15) and (6), was developed for the statistical inference, requiring less CPU time than state-of-the-art tools, while providing parallel scalability to deal with larger domains. Moreover, the new constrained learning algorithm for Bayesian networks yielded performance gains in predicting the most likely structure given new sentences (unseen during the training).

APPENDIX A

This appendix details our special formulation for the MAP optimization problem, whose fitness function is given by the joint probability:

$$P\left(x_{(-1,i)}, x_{(0,j)}, \ldots, x_{(N_i,k)} \mid f; \theta_{-1}, \ldots, \theta_{N_i}\right) = \frac{\prod_{q=-1}^{N_i} e^{\theta_q \phi_q\left(x_{(q,i)}, Pa_q, f\right)}}{\prod_{q=-1}^{N_i} \sum_{h=1}^{|S_q|} e^{\theta_q \phi_q\left(x_{(q,h)}, Pa_q, f\right)}} \tag{8}$$

where $x_{(q,i)}$ is the i^{th} discrete value of the variable X_q and Pa_q represents the state of the parents of node q. Therefore, since $log\left(\cdot\right)$ is a monotonically increasing function on R_+, the optimization task can be written as:

$$\left\{x_{-1}^*, x_0^*, \ldots, x_{N_i}^*\right\} = \underset{X_{-1}, X_0, \ldots, X_{N_i}}{\arg\max} \; \Pi \tag{9}$$

where

$$\Pi = \sum_{q=-1}^{N_i} \theta_q \phi_q\left(X_q, Pa_q, f\right) - \sum_{q=-1}^{N_i} \log \sum_{h=1}^{|S_q|} e^{\theta_q \phi_q\left(x_{(q,h)}, Pa_q, f\right)} \tag{10}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2017.2657690, IEEE Transactions on Computational Intelligence and AI in Games

The above optimization problem yields a search space, whose cardinality is $\prod_{q=-1}^{N_i} |S_q|$. Notice that it is possible to rewrite (9)-(10) as:

$$\left\{\bar{x}_1^*, \ldots, \bar{x}_{\bar{M}}^*, \widetilde{x}_1^*, \ldots, \widetilde{x}_{\widetilde{M}}^*\right\} = \underset{\bar{X}_1,\ldots,\bar{X}_{\bar{M}},\widetilde{X}_1,\ldots,\widetilde{X}_{\widetilde{M}}}{\arg\max} \Pi_1 + \Pi_2 \tag{11}$$

where

$$\begin{aligned}\Pi_1 = &\textstyle\sum_{q=1}^{\bar{M}} \bar{\theta}_q \bar{\phi}_q \left(\bar{X}_q, Pa_q, f\right) - \\ &\textstyle\sum_{q=1}^{\bar{M}} \log \sum_{h=1}^{|\bar{S}_q|} e^{\bar{\theta}_q \bar{\phi}_q\left(\bar{x}_{(q,h)}, Pa_q, f\right)}\end{aligned} \tag{12}$$

and

$$\begin{aligned}\Pi_2 = &\textstyle\sum_{j=1}^{\widetilde{M}} \widetilde{\theta}_j \widetilde{\phi}_j \left(\widetilde{X}_j, Pa_j, f\right) - \\ &\textstyle\sum_{j=1}^{\widetilde{M}} \log \sum_{h=1}^{|\widetilde{S}_j|} e^{\widetilde{\theta}_j \widetilde{\phi}_j\left(\widetilde{x}_{(j,h)}, Pa_j, f\right)}\end{aligned} \tag{13}$$

The search space of (11)-(13) has the same cardinality as (9)-(10), which can be rewritten as:

$$\prod_{q=1}^{\bar{M}} |\bar{S}_q| \prod_{j=1}^{\widetilde{M}} \left|\tilde{S}_j\right| \tag{14}$$

Also note that Π_2 is affected by the optimization of Π_1, since its parent nodes, Pa_j, belong to the set $\left\{\bar{X}_1, \ldots, \bar{X}_{\bar{M}}\right\}$. However, it is possible to exploit the conditional independence property of the Bayesian network, since the variables $\widetilde{X}_1, \ldots, \widetilde{X}_{\widetilde{M}}$ are conditionally independent given the values of $\bar{X}_1, \ldots, \bar{X}_{\bar{M}}$. For instance, in the case of our model it is possible to state that:

$$X_i \perp\!\!\!\perp X_j \mid X_0 \;\; \forall i, j \in \{1, \ldots, N_i\}, i \neq j \tag{15}$$

as can be seen in Fig. 2. The conditional independence enables the system to reduce the search space by carrying out the equivalent optimization problem:

$$\left\{\bar{x}_1^*, \ldots, \bar{x}_{\bar{M}}^*\right\} = \underset{\bar{X}_1,\ldots,\bar{X}_{\bar{M}}}{\arg\max} \Pi_1 + \Pi_2 \tag{16}$$

where

$$\begin{aligned}\Pi_1 = &\textstyle\sum_{q=1}^{\bar{M}} \bar{\theta}_q \bar{\phi}_q \left(\bar{X}_q, Pa_q, f\right) - \\ &\textstyle\sum_{q=1}^{\bar{M}} \log \sum_{h=1}^{|\bar{S}_q|} e^{\bar{\theta}_q \bar{\phi}_q\left(\bar{x}_{(q,h)}, Pa_q, f\right)}\end{aligned} \tag{17}$$

and

$$\begin{aligned}\Pi_2 = &\textstyle\sum_{j=1}^{\widetilde{M}} \widetilde{\theta}_j \widetilde{\phi}_j \left(\widetilde{x}_j^*, Pa_j, f\right) - \\ &\textstyle\sum_{j=1}^{\widetilde{M}} \log \sum_{h=1}^{|\widetilde{S}_j|} e^{\widetilde{\theta}_j \widetilde{\phi}_j\left(\widetilde{x}_{(j,h)}, Pa_j, f\right)}\end{aligned} \tag{18}$$

and $\widetilde{x}_j^*$ is found by solving the following sub-problem for $j = 1, \ldots, \widetilde{M}$:

$$\widetilde{x}_j^* = \underset{\widetilde{X}_j}{\arg\max}\, \widetilde{\theta}_j \widetilde{\phi}_j \left(\widetilde{X}_j, Pa_j, f\right) \tag{19}$$

The problem (16)-(19) exploits a small subspace of (9)-(10) of cardinality given by:

$$\prod_{q=1}^{\bar{M}} |\bar{S}_q| \sum_{j=1}^{\widetilde{M}} \left|\tilde{S}_j\right| \tag{20}$$

Although the optimization problem (16) only explicitly represents the variables, $\bar{X}_1, \ldots \bar{X}_{\bar{M}}$, at each iteration the algorithm stores the optimal values of $\widetilde{X}_1, \ldots, \widetilde{X}_{\widetilde{M}}$, resulting from the maximization (19), in order to provide the optimal instances of the whole set of variables.

APPENDIX B

This appendix details our new constrained learning method for Bayesian networks. Assuming that the training examples are independent and identically distributed (iid), it is possible to model the training of the statistical model (3) as the maximization of the joint probability:

$$\max_{\theta_{-1}\ldots\theta_{N_i}} \prod_{k=1}^{N_e} P\left(x_{(-1,k)}, \ldots, x_{(N_i,k)} \mid f_k; \theta_{-1}, \ldots, \theta_{N_i}\right) \tag{21}$$

where N_e is the cardinality of the training data set, $x_{(j,k)}$ is the target state of the j^{th} variable in the k^{th} semantic frame.

Since $log\left(\cdot\right)$ is a monotonically increasing function on R_+, the optimization task (21) is equivalent to:

$$\max_{\theta_{-1}\ldots\theta_{N_i}} \sum_{k=1}^{N_e} \log P\left(x_{(-1,k)} \ldots x_{(N_i,k)} \mid f_k; \theta_{-1} \ldots \theta_{N_i}\right) \tag{22}$$

Our constrained learning formulation replaces the usual training approach, (22), by the constrained optimization problem:

$$\begin{aligned}\min_{\theta_0\ldots\theta_M} \quad & -\sum_{k=1}^{N_e} \log P\left(x_{(-1,k)} \ldots x_{(N_i,k)} \mid f_k; \theta_{-1} \ldots \theta_{N_i}\right) \\ \text{s.t.} \quad & \xi \geq P\left(x_{(n,i)} \mid pa_{(n,k)}, f_k; \theta_n\right) \times \\ & P\left(x_{(m,j)} \mid pa_{(m,k)}, f_k; \theta_m\right) \left\{{}^{\forall k}_{\forall (x_{(n,i)}, x_{(m,j)}) \in I \times J}\right.\end{aligned} \tag{23}$$

where k is the index of the training example and $I \times J$ is a set of exclusivity constraints, in the form (7), defined by the user with the support of a user interface that makes it possible to define sub-sets of constraints at once for all the training examples, $k = 1, \ldots, N_e$. Substituting the expression of the adopted log-linear model into (23) and applying the logarithm on both sides of the constraint, yields:

$$\begin{aligned}\min_{\theta_{-1}\ldots\theta_{N_i}} \quad & -\rho\left(\theta_{-1} \ldots \theta_{N_i}\right) \\ \text{s.t.} \quad & \log \xi \geq \theta_m \phi_m \left(x_{(m,j)}, pa_{(m,k)}, f_k\right) - \\ & \log \sum_{h=1}^{|S_n|} \exp\left(\theta_n \phi_n \left(x_{(n,h)}, pa_{(n,k)}, f_k\right)\right) - \\ & \log \sum_{h=1}^{|S_m|} \exp\left(\theta_m \phi_m \left(x_{(m,h)}, pa_{(m,k)}, f_k\right)\right) + \\ & \theta_n \phi_n \left(x_{(n,i)}, pa_{(n,k)}, f_k\right) \left\{{}^{\forall k}_{\forall (x_{(n,i)}, x_{(m,j)}) \in I \times J}\right.\end{aligned} \tag{24}$$

where

$$\begin{aligned}&\rho\left(\theta_{-1} \ldots \theta_{N_i}\right) = \textstyle\sum_{k=1}^{N_e} \sum_{q=-1}^{N_i} \theta_q \phi_q \left(x_{(q,k)}, pa_{(q,k)}, f_k\right) - \\ &\textstyle\sum_{k=1}^{N_e} \sum_{q=-1}^{N_i} \log \sum_{h=1}^{|S_q|} \exp\left(\theta_q \phi_q \left(x_{(q,h)}, pa_{(q,k)}, f_k\right)\right)\end{aligned} \tag{25}$$

and S_n is the domain of the variable X_n, from which $x_{(n,h)} \in S_n$ is the h^{th} value belonging to the set S_n. Notice that (24) has a log-sum-exp term, originated from

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2017.2657690, IEEE Transactions on Computational Intelligence and AI in Games

the normalization of the probability distributions, which is repeated in the objective function, (25), and constraints. Therefore, to save computational effort, the above problem can be formulated in a more compact form as:

$$\begin{aligned}\min_{\theta_{-1}\dots\theta_{N_i}} \quad & -\sum_{k=1}^{N_e}\sum_{q=-1}^{N_i}\theta_q\phi_q\left(x_{(q,k)},pa_{(q,k)},f_k\right)\\ \text{s.t.} \quad & \log\beta \geq \theta_n\phi_n\left(x_{(n,i)},pa_{(n,k)},f_k\right)+\\ & \theta_m\phi_m\left(x_{(m,j)},pa_{(m,k)},f_k\right)\left\{{}^{\forall k}_{\forall(x_{(n,i)},x_{(m,j)})\in I\times J}\right.\\ & \log\sum_{h=1}^{|S_q|}\exp\left(\theta_q\phi_q\left(x_{(q,h)},pa_{(q,k)},f_k\right)\right)=0\left\{{}^{\forall k}_{\forall q}\right.\end{aligned} \tag{26}$$

where $\beta\in(0,1]$ is an upper bound, provided by the user, for the exclusivity constraints. The second constraint of (26) encodes the normalization, i.e. the denominator, of the log-linear model of the probability distribution, valid for both the objective function and exclusivity constraints. Notice that this constraint keeps the second term of (25) constant, while the objective function of (26) aims at increasing the first term of (25) (remembering that a minimization of a function multiplied by -1 is equivalent to its maximization). Therefore, both formulations maximize (25).

Unfortunately, (26) is not a convex problem, since an equality defines a convex domain if, and only if, it is an affine function, which is not the case of the second constraint of (26). However, it is also possible to maximize (25) by maximizing its first term, while bounding its second term, instead to keep it constant, as in (26). To do so, one can replace the equality of the second constraint of (26) by an inequality, while keeping the properties of a Bayesian model, since the likelihood given by (2) is normalized, obtaining a convex sub-normalized approximation of (26), as follows:

$$\begin{aligned}\min_{\theta_{-1}\dots\theta_{N_i}} \quad & -\sum_{k=1}^{N_e}\sum_{q=-1}^{N_i}\theta_q\phi_q\left(x_{(q,k)},pa_{(q,k)},f_k\right)\\ \text{s.t.} \quad & 0 \geq -\log\beta+\theta_n\phi_n\left(x_{(n,i)},pa_{(n,k)},f_k\right)+\\ & \theta_m\phi_m\left(x_{(m,j)},pa_{(m,k)},f_k\right)\left\{{}^{\forall k}_{\forall(x_{(n,i)},x_{(m,j)})\in I\times J}\right.\\ & \log\sum_{h=1}^{|S_q|}\exp\left(\theta_q\phi_q\left(x_{(q,h)},pa_{(q,k)},f_k\right)\right)\leq 0\left\{{}^{\forall k}_{\forall q}\right.\end{aligned} \tag{27}$$

The optimization problem (27) differs from (26) only by the equality constraint, which was replaced by an inequality, turning (26) into a convex optimization problem, since both the objective function and the first constraint of (27) are compositions of affine functions, being convex, while the second constraint is a log-sum-exp function, better known as a convex function. However, the second constraint of (27) makes the model sub-normalized, which isn't a problem, since the likelihood given by the log-linear model has a normalization term in the denominator.

Our framework offers two algorithms to solve (27), the interior point and the active set algorithms. To improve the precision and speed up the optimization, it is provided the partial derivatives of the objective function, henceforward called F, given by:

$$\frac{\delta F}{\delta\theta_q}=-\sum_{k=1}^{N_e}\phi_q\left(x_{(q,k)},pa_{(q,k)},f_k\right) \tag{28}$$

for $q=-1,\dots,N_i$. Since the objective function is linear, the derivatives are constant for any θ, so they are calculated only once, before calling the optimization algorithm.

ACKNOWLEDGMENT

This work was funded by the EU ICT FP7 FET project Machine Understanding for interactive StorytElling (MUSE) http://www.muse-project.eu/.

REFERENCES

[1] Martha Palmer, Daniel Gildea, and Nianwen Xue. Semantic role labeling. *Synthesis Lectures on Human Language Technologies*, 3(1):1–103, 2010.

[2] Charu C Aggarwal and Cheng Xiang Zhai. *Mining Text Data*. Springer Science & Business Media, 2012.

[3] Martha Palmer, Daniel Gildea, and Paul Kingsbury. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106, 2005.

[4] Collin F Baker, Charles J Fillmore, and John B Lowe. The berkeley framenet project. In *Proceedings of the 17th International Conference on Computational Linguistics-Volume 1*, pages 86–90, 1998.

[5] Xavier Carreras and Lluís Màrquez. Introduction to the conll-2005 shared task: Semantic role labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, pages 152–164, 2005.

[6] Vincenzo Lombardo and Antonio Pizzo. Ontology based visualization of characters intentions. In Alex Mitchell, Clara Fernndez-Vara, and David Thue, editors, *Interactive Storytelling*, volume 8832 of *Lecture Notes in Computer Science*, pages 176–187. 2014.

[7] Ruqian Lu and Songmao Zhang. *Automatic generation of computer animation: using AI for movie animation*. Springer-Verlag, 2002.

[8] Roger C Schank and Robert P Abelson. *Scripts, plans, goals, and understanding: An inquiry into human knowledge structures*. Psychology Press, 2013.

[9] Marc Verhagen, Roser Sauri, Tommaso Caselli, and James Pustejovsky. Semeval-2010 task 13: Tempeval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 57–62, 2010.

[10] Parisa Kordjamshidi, Marie-Francine Moens, and Martijn van Otterlo. Spatial role labeling: Task definition and annotation scheme. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, pages 413–420, 2010.

[11] Karl Pichotta and Raymond J Mooney. Learning statistical scripts with lstm recurrent neural networks. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, 2015.

[12] Josep Valls-Vargas, Santiago Ontanón, and Jichen Zhu. Toward character role assignment for natural language stories. In *Proceedings of the Ninth Artificial Intelligence and Interactive Digital Entertainment Conference*, pages 101–104, 2013.

[13] Vladimir IAkovlevich Propp. *Morphology of the Folktale*, volume 9. American Folklore Society, 1958.

[14] Wendy G Lohnert, John B Black, and Brian J Reiser. Summarizing narratives. In *Proceedings of the 7th International Joint Conference on Artificial intelligence-Volume 1*, pages 184–189. Morgan Kaufmann Publishers Inc., 1981.

[15] Amit Goyal, Ellen Riloff, and Hal Daumé III. Automatically producing plot unit representations for narrative text. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 77–86, 2010.

[16] Mark Alan Finlayson. *Learning narrative structure from annotated folktales*. PhD thesis, Massachusetts Institute of Technology, 2012.

[17] Seung Y Lee, Jonathan P Rowe, Bradford W Mott, and James C Lester. A supervised learning framework for modeling director agent strategies in educational interactive narrative. *Computational Intelligence and AI in Games, IEEE Transactions on*, 6(2):203–215, 2014.

[18] Quynh Ngoc Thi Do, Steven Bethard, and Marie-Francine Moens. Adapting coreference resolution for narrative processing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Lisboa, Portugal, 2015.

[19] Ronald Melzack and Carol Jones. *The Day Tuk Became a Hunter & Other Eskimo Stories*. Dodd, Mead New York, 1967.

[20] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[21] Eric Bigelow, Daniel Scarafoni, Lenhart Schubert, and Alex Wilson. On the need for imagistic modeling in story understanding. *Biologically Inspired Cognitive Architectures*, 11:22–28, 2015.

[22] Oswaldo Ludwig, Xiao Liu, Parisa Kordjamshidi, and Marie-Francine Moens. Deep embedding for spatial role labeling. *arXiv preprint arXiv:1603.08474*, 2016.

[23] Christos T Rodosthenous, Loizos Michael, Mark A Finlayson, Jan Christoph Meister, and Emile G Bruneau. Gathering background knowledge for story understanding through crowdsourcing. In *Proceedings of the 5th Workshop on Computational Models of Narrative (CMN 2014)*, volume 41, pages 154–163, 2014.

[24] Parag Singla and Pedro Domingos. Entity resolution with markov logic. In *Data Mining, 2006. ICDM'06. Sixth International Conference on*, pages 572–582. IEEE, 2006.

[25] Thorsten Joachims, Thomas Finley, and Chun-Nam John Yu. Cutting-plane training of structural SVMs. *Machine Learning*, 77(1):27–59, 2009.

[26] Paul Kingsbury and Martha Palmer. Propbank: the next level of treebank. In *Proceedings of Treebanks and Lexical Theories*, volume 3, 2003.

[27] Solomon Eyal Shimony. Finding maps for belief networks is np-hard. *Artificial Intelligence*, 68(2):399–410, 1994.

[28] David Sontag and Tommi Jaakkola. Tree block coordinate descent for map in graphical models. *Journal of Machine Learning Research*, 5:544–551, 2009.

[29] László Lovász. On the ratio of optimal integral and fractional covers. *Discrete mathematics*, 13(4):383–390, 1975.

[30] Azmi Alazzam and Harold W Lewis III. A new optimization algorithm for combinatorial problems. *IJARAI) International Journal of Advanced Research in Artificial Intelligence*, 2(5), 2013.

[31] Oswaldo Ludwig, Pablo Gonzalez, and Antonio Cezar Lima. Optimization of ANN applied to non-linear system identification. In *Proceedings of the 25th IASTED international conference on Modeling, indentification, and control*, pages 402–407, Anaheim, CA, USA, 2006. ACTA Press.

[32] Oswaldo Ludwig and Urbano Nunes. Novel maximum-margin training algorithms for supervised neural networks. *Neural Networks, IEEE Transactions on*, 21(6):972 –984, jun. 2010.

[33] Oswaldo Ludwig, Urbano Nunes, and Rui Araujo. Eigenvalue decay: A new method for neural network regularization. *Neurocomputing*, 124:33–42, 2014.

[34] Elahe Rahimtoroghi, Thomas Corcoran, Reid Swanson, Marilyn A Walker, Kenji Sagae, and Andrew Gordon. Minimal narrative annotation schemes and their applications. In *Seventh Intelligent Narrative Technologies Workshop*, 2014.