



Combining Deep Learning and Symbolic Processing for Extracting Knowledge from Raw Text

Andrea Zugarini^{1,3}, Jérémy Morvan², Stefano Melacci^{3(✉)}, Stefan Knerr⁴,
and Marco Gori³

¹ DINFO, University of Florence, Florence, Italy
andrea.zugarini@unifi.it

² Criteo, Paris, France
morvanjeremy@gmail.com

³ DIISM, University of Siena, Siena, Italy
{mela,marco}@diism.unisi.it

⁴ CogniTalk, Nantes, France
stefan.knerr@cognitalk.com

Abstract. This paper faces the problem of extracting knowledge from raw text. We present a deep architecture in the framework of Learning from Constraints [5] that is trained to identify mentions to entities and relations belonging to a given ontology. Each input word is encoded into two latent representations with different coverage of the local context, that are exploited to predict the type of entity and of relation to which the word belongs. Our model combines an entropy-based regularizer and a set of First-Order Logic formulas that bridge the predictions on entity and relation types accordingly to the ontology structure. As a result, the system generates symbolic descriptions of the raw text that are interpretable and well-suited to attach human-level knowledge. We evaluate the model on a dataset composed of sentences about simple facts, that we make publicly available. The proposed system can efficiently learn to discover mentions with very few human supervisions and that the relation to knowledge in the form of logic constraints improves the quality of the system predictions.

Keywords: Information Extraction · Learning from Constraints
Deep Learning · Symbolic knowledge representation

1 Introduction

Information Extraction (IE) is one of the most important fields in Natural Language Processing (NLP), and it is about extracting structured knowledge from unstructured text [17]. IE encompasses a large variety of sub-problems, and, for

J. Morvan—Word done while at CogniTalk, Nantes, France.

the purpose of this work, we mostly consider Named Entity Recognition (NER) and Relation Extraction (RE).

The goal of NER systems is to detect and classify proper nouns according to a predefined set of entity types, such as “Person”, “Organization”, “Location”, and others. Many NER systems [10, 18] rely on handcrafted features and external knowledge, such as gazetteers or capitalization information. On one hand, they can help in spotting named entities, but, on the other hand, these techniques are usually tied to the considered task. Differently, Collobert et al. [4] deeply studied a neural model requiring only minor feature engineering. Their model was applied to several NLP problems, such as part-of-speech tagging, chunking, semantic role labeling, NER, and language modeling. More recent approaches make a wide use of recurrent neural networks (mostly LSTMs [6]), such as the one of Lample et al. [8], Chiu and Nichols [2] exploited similar networks, but character-level features are detected by convolutional nets, also used in [20].

Relation Extraction addresses the problem of finding and categorizing relations between entities in a given text document. This problem is even harder than NER, since relations are expressed in much more ambiguous ways than entity names. There is also a big issue related to RE, that is the lack of large collections of high quality labeled data. Relations can be implicit, they can have fuzzy boundaries, and they can also be constituted of non-contiguous words. Labeling can be hard even for humans, and it can be strongly inconsistent among different supervisors. Some approaches rely only on unsupervised models [16, 19], segmenting the word sequences (“mentions”) bounded by two defined entities. Mintz et al. [13] proposed an alternative paradigm, the so called “distant supervision”, that is a simple form of weak supervision. Intuitively, the distant approach is founded on the idea that sentences containing the same pair of entities are likely to express the same relation. Entities are taken from Freebase¹, and the considered relations are the ones that link the entity pair in the knowledge base. Miwa and Bansal [15] presented an end-to-end solution to extract both relations and entities from sentences. Their approach is based on stacked bidirectional tree structured LSTMs, where entities are extracted first, then relations are predicted.

This review shows that Deep Learning achieved serious improvements in NLP and IE-related applications. The renewed interest in recurrent neural networks and the introduction of distributed representations of words and sentences [1, 4, 12] allowed researchers to construct several systems that can be trained end-to-end, removing the costly efforts in feature engineering. However, these methods require large amounts of data to work properly, that in most of the cases need to be labeled. Supervisions are expensive, and, in the specific case of IE, researchers tend to focus on precise sub-tasks that are well studied and defined. Some of them (e.g. NER and RE) share several aspects, and addressing those problems jointly can be fruitful.

This work faces the problem of linking text portions to a given ontology with a known schema that is composed of entity and relation types. NER and RE can

¹ <https://developers.google.com/freebase/>.

be seen as special instances of the considered setting. The problems of recognizing and segmenting mentions to both entity and relation types are treated in a uniform way, reformulating them as problems of making predictions on a word given its context. While predicting the entity type of the mention to which the word belongs usually requires just a local context, spotting the relation type in which the word is involved needs a larger context. Following this intuition, we propose the idea that every word in the sentence can be considered from two different perspectives that we refer to as “narrow” and “broad” views. We propose a deep architecture that develops two latent representations of each word, associated to the just mentioned views. A predictor of entity types is attached to the former representation, whereas a relation type prediction operates on the latter. Our architecture is an instance of the generic framework of Learning from Constraints [5], where the unifying notion of “constraint” is used to inject knowledge coming from supervised and unsupervised data as well. In particular, an entropy-based index (that resembles the mutual information from the input views to the predictors) is maximized over all the data that is read by the system (labeled or not), while First-Order Logic (FOL) formulas are used to bridge predictions of entity and relation types. Formulas are converted into constraints by means of T-Norms. Linking the predictions on the two views allows the system to mutually improve their quality, differently from those models that treat them independently. When tested on a collection of sentences about factual knowledge, our model achieves good performances without requiring a large number of supervisions. This becomes more evident when logic constraints are introduced between the two views. We notice that this approach allows us to build neural models that provide an interpretable description of the unstructured raw text, by means of the FOL formalism. This interpretability, that is usually missing in neural architectures, offers a suitable basis to easily introduce additional information provided by an external supervisor. As a matter of fact, having a human-in-the-loop is known to be a crucial element in those models that learn and expand their internal knowledge bases in a life-long-learning setting [14].

This paper is organized as follows. Section 2 describes the proposed architecture and the logic constraints. Section 3 reports our experimental results, while Sect. 4 concludes the paper.

2 Model

We are given a data collection \mathcal{D} composed of b utterances. Every utterance $u \in \mathcal{D}$ consists of $|u|$ words indicated with $w_j, \forall j = 1, \dots, |u|$. We are also given an ontology \mathcal{O} , composed of k_n entity types and k_b relation types. Relations involve pairs of entities of pre-defined types, as sketched in Fig. 1. For each word w_i , the goal of our model is to learn to predict what is the entity type associated to word w_i , and what is the relation type to which w_i participates. For example, in the sentence *Paris is the capital of France*, the system should predict that *Paris* is an entity of type “city”, that *France* is an entity of type “country”, and that each word of the sentence is associated to the relation type “capitalof”, where all the mentioned types belong to the given ontology.

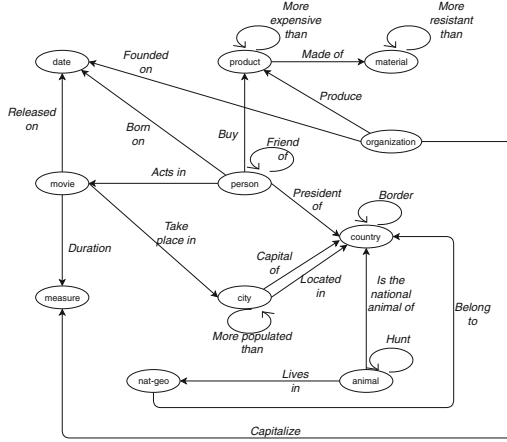


Fig. 1. Ontology. Nodes are entity types, while edges are relation types.

We follow the idea of developing two latent representations of w_i , that we refer to as “narrow” and “broad” views, respectively. We indicate such representations as $x_i^{(n)} \in \mathcal{X}^{(n)}$ and $x_i^{(b)} \in \mathcal{X}^{(b)}$, where $\mathcal{X}^{(n)}, \mathcal{X}^{(b)}$ are the generic spaces to which they belong, and n, b stand for “narrow” and “broad”. Determining the entity type of w_i can be usually done by considering a local view around it, and that is what $x_i^{(n)}$ encodes. Finding the relation type of w_i usually requires to have a wider view around w_i , since mentions to relations involve larger spans of text, that is the rationale behind representation $x_i^{(b)}$.

We consider a fixed-size vocabulary of words \mathcal{V} , so that each w_j is a 1-hot representation of size $|\mathcal{V}|$, and those w_j that are not covered by \mathcal{V} are marked with a generic symbol **unk**. Computing each $x_i^{(\cdot)}$ (being it narrow or broad) is the outcome of two main computational stages. The first stage consists in projecting the target symbol w_i into a latent (distributed) representation $e_i \in \mathbb{R}^d$, where d is the dimensionality of the embedding space. The embeddings of our vocabulary $\{e_i, i = 1, \dots, |\mathcal{V}|\}$, are stored (column-wise) into $\mathcal{W} \subset \mathbb{R}^{|\mathcal{V}| \times d}$, so that

$$e_i = \mathcal{W}w_i. \quad (1)$$

The second stage consists in fusing the distributed representations of the target word itself and of the other words around it, thus generating $x_i^{(\cdot)}$ by means of a Bidirectional Recurrent Neural Network (Bi-RNN). In detail, the Bi-RNN is composed by two RNNs that process two sequences of word embeddings,

$$\begin{aligned} S^{\rightarrow} &= e_1, e_2, \dots, e_i \\ S^{\leftarrow} &= e_{|u|}, e_{|u|-1}, \dots, e_i. \end{aligned}$$

Both sequences terminate in the position of the target word, but S^{\rightarrow} starts from the beginning of the sentence, while S^{\leftarrow} starts from the end. Hidden states of

the RNNs at the end of the sequences are concatenated, generating $x_i^{(\cdot)}$. We use Long Short Term Memories (LSTMs) to implement each RNN,

$$\begin{aligned} x_i^{(n)} &= [LSTM^{(n\rightarrow)}(S^{\rightarrow}), LSTM^{(n\leftarrow)}(S^{\leftarrow})] \\ x_i^{(b)} &= [LSTM^{(b\rightarrow)}(S^{\rightarrow}), LSTM^{(b\leftarrow)}(S^{\leftarrow})] . \end{aligned}$$

The global architecture is depicted in Fig. 2. While the embeddings $\{e_i\}$, are shared by the “narrow” and “broad” paths, the Bidirectional RNNs are independently developed in the two views. We implement the broad path of Fig. 2 (right side) by stacking multiple layers of Bidirectional RNNs. The last layer will embrace a larger context (due to the compositional effects of the stacking process), and it will model a higher-level/coarser representation of the input word. We notice that, in general, the broad representation could embrace multiple sentences, or even paragraphs.

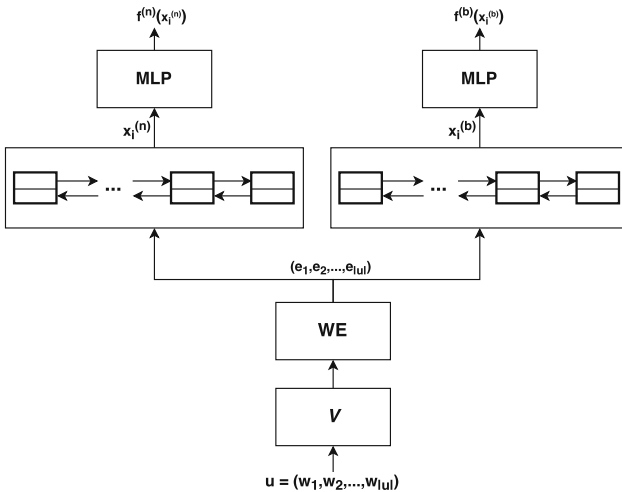


Fig. 2. Architecture of the proposed model. The utterance u is converted in a sequence of embeddings $e_1 \dots, e_{|u|}$, feeding two bidirectional LSTMs, that compute two representations $(x_i^{(n)}, x_i^{(b)})$ of each word w_i , also referred to as “narrow” and “broad” views (left and right paths, respectively, where the right path usually includes multiple layers of LSTMs). The predictor (MLP) on the “narrow” view outputs the entity type to which w_i belong, while the MLP on the “broad” view is about the relation type of w_i .

For each word w_i , we make predictions on the entity/relation types of the ontology \mathcal{O} that are more compatible with w_i . In particular, we introduce two set of functions that model the classifiers of each entity/relation type of \mathcal{O} ,

$$f^{(n)} = [f_1^{(n)}, \dots, f_{k_n}^{(n)}] : \mathcal{X}^n \rightarrow [0, 1]^{k_n} \quad (2)$$

$$f^{(b)} = [f_1^{(b)}, \dots, f_{k_b}^{(b)}] : \mathcal{X}^b \rightarrow [0, 1]^{k_b} \quad (3)$$

where $f^{(n)}$ are about entities and $f^{(b)}$ are about relations, and every component of vectors $f^{(n)}$ and $f^{(b)}$ is in the range $[0, 1]$. Both $f^{(n)}$ and $f^{(b)}$ are multilayer perceptrons having k_n and k_b output units with sigmoidal activations, as can be observed in the upper portions of the architecture in Fig. 2.

2.1 Semantic Features

Since predictors in (2) and (3) are associated to interpretable entity/relation types of the ontology \mathcal{O} , we will also refer to them as “semantic features”.

We expect semantic features to be learned from data by enforcing a combination of several constraints (they are enforced in a soft manner, so they are implemented as penalty functions to be minimized - we keep using the generic term “constraint” when referring to them). Each constraint guides the learning process accordingly to a specific principle that we describe in what follows. The objective function of our problem is Z , and we seek for those $f^{(n)}, f^{(b)}$ for which Z is minimal,

$$\min_{f^{(n)}, f^{(b)}} Z \left(f^{(n)}, f^{(b)} \right) = \min_{f^{(n)}, f^{(b)}} \sum_{f \in \{f^{(n)}, f^{(b)}\}} [R(f) + C(f, \mathcal{L}) + U(f)] + \Phi \left(f^{(n)}, f^{(b)} \right). \quad (4)$$

where $R(f)$ is a regularization term (implemented with the classical weight decay approach). The term $C(f, \mathcal{L})$ is the traditional cross-entropy loss, commonly used in supervised learning, that enforces a supervision constraint on those words that have been supervised by an expert, collected (together with the supervision) in \mathcal{L} . Not all the words are supervised, and in the experiments we will also evaluate the case in which no-supervisions are provided at all. The term $U(f)$, is a constraint inspired by the idea of maximizing the mutual information from the space of word representations to the space of semantic features [11],

$$U(f) = \sum_{s=1}^b \left[\sum_{i=1}^{|u_s|} P(f(x_{si}), \theta_v) + P(\max_p f(x_{si}), \theta_h) \right] + \lambda_g \cdot G(f). \quad (5)$$

where s is the index of a sentence in \mathcal{D} , while i is the word index. In detail, $U(f)$ is a sum of two contributions: the one in square brackets enforces the development of only a small number of features on each word/sentence, while $G(f)$ ensures an unbiased development of the features over all the dataset \mathcal{D} . The sets θ_v, θ_h collect some customizable positive scalars ($U(f)$ in Eq. (4) is applied to narrow and broad features independently, so we have two independent pairs of θ_v, θ_h), while λ_g is a tuneable weight > 0 . In detail, if $H(v) = -\sum_{k=1}^{|v|} v_k \log v_k$ is the Shannon entropy, we have

$$G(f) = -H \left(\frac{1}{b} \sum_{s=1}^b \frac{1}{|u_s|} \sum_{i=1}^{|u_s|} f(x_{si}) \right) \quad (6)$$

$$P(v, \theta = \{\lambda_1, \lambda_2, \gamma\}) = \lambda_1 \cdot H(v) + \lambda_2 \left(\sum_{k=1}^{|v|} v_k - \gamma \right)^2. \quad (7)$$

The term $G(f)$ is minimized when all the semantic features are activated uniformly over all \mathcal{D} , on average. The loss P is minimized when the activations provided as its first argument are “close” to 1-hot vectors, that means that we want a few-strongly-activated elements. Notice that the term P is used twice in (5). Its first instance applies to the activations of semantic features on each “word”. The second instance is about the activations of semantic features on each “sentence” (pooled with the max operator). As a matter of fact, P constraints our model to develop a few, well activated features on each word, and a few well activated features on each sentence. Both the terms G and P involve the entropy function H , that is meaningful when its input is a probability distribution. For this reason, the squared term in P introduces a competition among the provided activations, that are enforced to sum to γ . If $\gamma = 1$ we have a probability distribution, while if $\gamma > 1$ (but still small) we have a sort of relaxed implementation of the probabilistic relationships. This makes the system more tolerant to multiple activations on the same input, that, from the practical point of view, turns out to be desirable. The last term of (4), $\Phi(f^{(n)}, f^{(b)})$, is a constraint coming from First-Order Logic (FOL), that introduces a link between the two views on the data, with the goal of improving the quality of both the categories of semantic features.

2.2 Logic Constraints

Narrow and broad semantic features are related to each other due to their link in the ontology \mathcal{O} . Consider, for example, an ontology composed of entity types “city”, “country”, and of the relation type “capitalof”. In the following sentence, *Paris is the capital of France*, we can associate the entity type “city” to *Paris*, the type “country” to *France* and the relation type “capitalof” to each word of the sentence (since they all contribute to such relation). Our system is expected to give strong activation to the features indicate below (for the purpose of this description, we make explicit the entity/relation type to which each feature is associated),

$$\overbrace{\underbrace{Paris}_{f_{city}^{(n)}} \text{ is the capital of } \underbrace{France}_{f_{country}^{(n)}}}_{f_{capitalof}^{(b)}}.$$

We can clearly grasp that whenever the narrow features $f_{city}^{(n)}$ and $f_{country}^{(n)}$ are active together in a sentence, it is very likely that the sentence involves the relation “being a capital of”, i.e., that $f_{capitalof}^{(b)}$ should be active too, and vice-versa¹. Since the functions f model the activation of predicates of the ontology \mathcal{O} , we can implement this relationship by means of FOL formulas, such as

¹ In general, this could be ambiguous, since multiple relations could be associated to a city and a country. We solve this problem by introducing a special narrow feature for each broad function (to simplify the presentation, we avoid going into further details).

$$f_{city}^{(n)} \wedge f_{country}^{(n)} \Leftrightarrow f_{capitalof}^{(b)} .^1 \quad (8)$$

We repeat this process for each relation in \mathcal{O} , getting multiple formulas, that are then translated into real-valued constraints $\phi_r(f^{(n)}, f^{(b)})$ by means of T-Norms, as studied in [5]. For each T-Norm, there exists a unique function \Rightarrow called residuum, satisfying certain properties which is the natural translation of the logic implication. In this work, we considered the residuum of the *Lukasiewicz* T-Norm. Łukasiewicz logic presents good properties such as the involutive negation. However, translating a large chain of \wedge operations with such T-Norm requires a strong activation of all the involved components because the sum of n features should be greater than $n - 1$. This could be sometimes a strong requirement to satisfy. Hence, we converted the \wedge operator using the Gödel T-Norm, which instead defines such operator as the minimum among the whole predicates,

$$f_1 \wedge f_2 \wedge \dots \wedge f_n = \min(f_1, f_2, \dots, f_n) . \quad (9)$$

Departing from the provided example, in the ontology \mathcal{O} we have a large number of relations and, for each of them, we can build a FOL formula as (8), and translate it into a real-valued penalty function. Summing up all the penalties, we get

$$\Phi(f^{(n)}, f^{(b)}) = \lambda_l \cdot \sum_{r=1}^{k_b} \phi_r(f^{(n)}, f^{(b)}), \quad (10)$$

where $\lambda_l > 0$ is a customizable scalar. We remark that whenever the activations of the premises and of the conclusions of (8) are both small (i.e., false), the corresponding constraints are automatically satisfied. The actual contribution of each $\phi_r(f^{(n)}, f^{(b)})$ becomes significant whenever there is disagreement between the semantic features computed on the broad and narrow sides.

2.3 Segmentation

In Eq. (8) we did not make explicit the arguments on which semantic features operate. While semantic features are designed to make predictions on single words, the FOL constraints can involve longer portions of text, uniformly referred to as “segments” (the previous example involved two single-word segments for narrow features - *Paris*; *France* - but, in general, we can have longer segments). In order to evaluate the FOL constraints we need to segment the input sentence and compute segment-level activations of the semantic features.

Segmentation is performed as follows: for each word, we mark as “active” only the narrow features whose activation score is beyond a decision threshold (assumed to be 0.5). If multiple nearby words share the same active feature, we collapse them into a single segment. This procedure generates multiple segmentation hypotheses for each narrow feature. We prune the hypotheses by keeping only the segment with the strongest activation (we kept also a second hypothesis for those narrow features involved twice in the same relation). In the case

of broad features, for simplicity, we assume that there is only a single segment that cover the whole sentence. Finally, segment-level activations are computed by averaging the word-level scores. An example of segment activation/selection procedure is illustrated in Table 1 (the first entity is badly segmented).

Table 1. Segment generation/selection. Shaded elements are above the activation threshold (0.5), whereas bordered rectangles indicates the segments we select.

	Paris	is	the	capital	of	France	Avg Score	Activated
$f_{city}^{(n)}$	0.95	0.71	0.2	0.14	0	0.3	0.83	✓
$f_{country}^{(n)}$	0.5	0.65	0.05	0	0.1	0.9	0.9	✓
$f_{capitalof}^{(b)}$	0.35	0.7	0.9	0.8	0.8	0.1	0.6	✓

3 Experiments

We manually created (and made public) a dataset² \mathcal{D} with sentences that are word-by-word linked to a given ontology. The dataset is a collection of 1000 sentences, where 700 are used to train our model and 300 are used as test set. Each sentence is constituted by a triple structured as *entity1-relation-entity2*. Our ontology is composed of 11 entity types, and 23 relation types, whose organization is exactly the one that we have already shown in Fig. 1, where nodes are entity types and links are relation types. We kept data simple; sentences have no co-references, quite explicit relation expressions, and the vocabulary \mathcal{V} covers almost all the words. We intentionally introduced some noise in the labeling process, to make the task more challenging.

Word embeddings \mathcal{W} were initialized (and kept fixed) with the ones from [3], that are vectors with 50 components. The sizes of recurrent network states (500) and the hidden layers of multilayer perceptron blocks have been chosen by cross-validation. In the case of broad features we have two layers of recurrence. The narrow and broad MLP-based predictors have a single layer with 1200 and 800 hidden units, respectively. Our cost function was optimized with ADAM [7], using mini-batches of size 32 (sentences), and we also introduced some gradient noise and gradient clipping.

The objective function in Eq. 4 requires the tuning of several hyper-parameters. However, the values of the parameter γ in the sets θ_v and θ_h can be defined exploiting prior knowledge on the sentence structure (recall that we have two independent pairs (θ_v, θ_h) for broad and narrow features). Broad features are supposed to be 1-hot in each word, and the active broad feature should be the same in the whole sentence. Thus, we set $\gamma = 1$ in both θ_v and θ_h . Likewise, we expect only one narrow feature per word, which means $\gamma = 1$ in the case of θ_v , but here the number of features per sentence is set to $\gamma = 3$ in the case of θ_h . The remaining hyper-parameters were tuned by cross-validation.

² <http://sailab.diism.unisi.it/onto-sentences-dataset/>.

Experiments are evaluated with different scores. A common metric used in Information Extraction is the F_1 score, but it can be only applied with labeled data, so it is not possible to measure entropy-based constraints when enforced in a completely unsupervised framework. In such situations we evaluate the mutual information between the semantic features and the ground truth. In particular, we adopted the Normalized Mutual Information (NMI) [9].

We consider a sparsely supervised setting and we compare our model against two simplified instances of it: one is trained only using constraints on the supervised examples, i.e. without entropy-based and logic constraints, another one exploits also entropy-based constraints but not logic formulas. We varied the number of labeled sentences in the training set ranging from only 1 supervised sentence per relation type, to a fully labeled case (“all”). Additionally, one of the models is also trained without considering any supervised data at all.

Figure 3 reports our results. First, we focus on the scores obtained in the case in which supervised constraints are not exploited. Since we are in a fully unsupervised case, we do not introduce logic constraints, so that only one plot is meaningful (green line, first dot of the plot). This is due to the fact that in the unsupervised case we do not have access to the symbolic elements of the ontology that are associated to the semantic features. The NMI scores in the narrow and broad cases (Fig. 3 (a, c)) show that although entropy constraints produce a significant score in the case of broad features, the result on narrow features are not encouraging. As a matter of fact, words in the borders of two entity types are sources of errors. In the case of broad features, since we output a prediction on the whole sentence, this issue is not present.

When supervised examples are introduced, Fig. 3 (a) shows that even only one supervised sentence per formula remarkably improves the NMI score of narrow features. Interestingly, the unsupervised case, despite its low performances, is still better than using a single supervision. Differently, broad features are less affected by the introduction of the first supervised example (Fig. 3 (c)), since they were already showing good performances in the fully unsupervised case. Performances of semi-supervised models (both in the case of entropy and entropy + logic) are significantly better than the model trained only with supervisions (NMI and F1, Fig. 3 (a, b, c, d)). More generally, the entropy-based terms are crucial whenever the number of supervised data is limited. Only when we go beyond 10 supervised sentences per formula (\approx one third of the training set) the supervised case gets closer to the semi-supervised entropy-based case, but still does not reach the case in which logic formulas are added. Introducing logic formulas almost constantly gives improvements over the entropy-only case, confirming that bridging the predictions on broad and narrow views is important to allow a positive transfer of information between the two views.

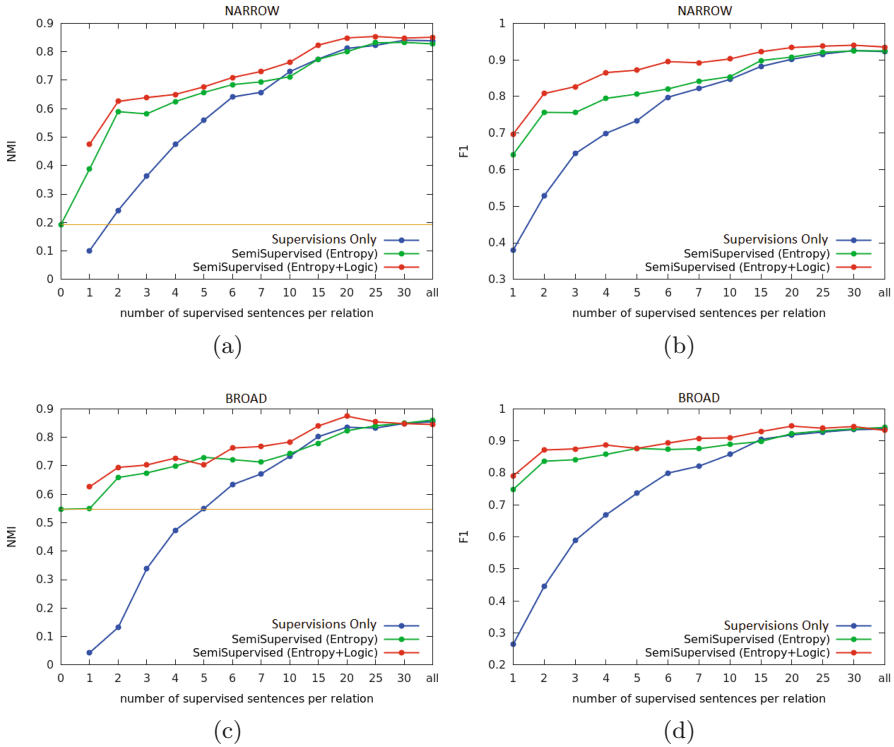


Fig. 3. Comparison of the quality of semantic features on narrow (a), (b) and broad (c), (d) views in the case of different models (test set). While (a), (c) are about the NMI score, (b), (d) report the F1 measure. The entropy-based constraints (green curve, NMI only, (a), (c)) are also evaluated in the unsupervised case (the yellow line repeats this result over all the graph, as reference). (Color figure online)

4 Conclusions

We presented a deep architecture in the framework of Learning from Constraints [5], that was designed to extract and identify mentions to entity and relation types belonging to a given ontology. Thanks to the introduction of two latent representations (views) of the input data, we implemented entity and relation detectors in a uniform way, differently from several existing systems. Our results have shown that introducing ontology-related information, represented as First-Order Logic formulas, helps the system to improve the quality of its predictions.

Our model must be extended to larger scale data and evaluated in less controlled environments. We plan to investigate more challenging settings, following the idea of life-long learning, and departing from the usual batch-mode approach toward a framework where there is an online interaction with humans. This is made possible by the interpretable representations of the raw text that are generated by our model.

References

1. Bengio, Y., Ducharme, R., Vincent, P., Jauvin, C.: A neural probabilistic language model. *J. Mach. Learn. Res.* **3**(Feb), 1137–1155 (2003)
2. Chiu, J., Nichols, E.: Named entity recognition with bidirectional LSTM-CNNs. *Trans. Assoc. Comput. Linguist.* **4**(1), 357–370 (2016)
3. Collobert, R., Weston, J.: A unified architecture for natural language processing: deep neural networks with multitask learning. In: *Proceedings of the 25th International Conference on Machine Learning*, pp. 160–167. ACM (2008)
4. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. *J. Mach. Learn. Res.* **12**(Aug), 2493–2537 (2011)
5. Gnecco, G., Gori, M., Melacci, S., Sanguineti, M.: Foundations of support constraint machines. *Neural Comput.* **27**(2), 388–480 (2015)
6. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* **9**(8), 1735–1780 (1997)
7. Kingma, D., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint [arXiv:1412.6980](https://arxiv.org/abs/1412.6980) (2014)
8. Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., Dyer, C.: Neural architectures for named entity recognition. In: *Proceedings of NAACL-HLT*, pp. 260–270 (2016)
9. Lancichinetti, A., Fortunato, S., Kertész, J.: Detecting the overlapping and hierarchical community structure in complex networks. *New J. Phys.* **11**(3), 033015 (2009)
10. Luo, G., Huang, X., Lin, C.Y., Nie, Z.: Joint named entity recognition and disambiguation. In: *Proceedings EMNLP* (2015)
11. Melacci, S., Gori, M.: Unsupervised learning by minimal entropy encoding. *IEEE Trans. Neural Netw. Learn. Syst.* **23**(12), 1849–1861 (2012)
12. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: *Advances in Neural Information Processing Systems*, pp. 3111–3119 (2013)
13. Mintz, M., Bills, S., Snow, R., Jurafsky, D.: Distant supervision for relation extraction without labeled data. In: *Proceedings of ACL*, pp. 1003–1011 (2009)
14. Mitchell, T., Cohen, W., Hruschka, E., Talukdar, P., Yang, B., et al.: Never-ending learning. *Commun. ACM* **61**(5), 103–115 (2018)
15. Miwa, M., Bansal, M.: End-to-end relation extraction using LSTMs on sequences and tree structures. arXiv preprint [arXiv:1601.00770](https://arxiv.org/abs/1601.00770) (2016)
16. Pantel, P., Bhagat, R., Coppola, B., Chklovski, T., Hovy, E.H.: ISP: learning inferential selectional preferences. In: *HLT-NAACL*, pp. 564–571 (2007)
17. Piskorski, J., Yangarber, R.: Information extraction: past, present and future. In: Poibeau, T., Saggion, H., Piskorski, J., Yangarber, R. (eds.) *Multi-Source, Multilingual Information Extraction And Summarization. NLP*, pp. 23–49. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-28569-1_2
18. Ratinov, L., Roth, D.: Design challenges and misconceptions in named entity recognition. In: *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pp. 147–155. Association for Computational Linguistics (2009)
19. Shinyama, Y., Sekine, S.: Preemptive information extraction using unrestricted relation discovery. In: *Proceedings of NAACL HLT*, pp. 304–311 (2006)
20. Strubell, E., Verga, P., Belanger, D., McCallum, A.: Fast and accurate sequence labeling with iterated dilated convolutions. arXiv preprint [arXiv:1702.02098](https://arxiv.org/abs/1702.02098) (2017)