

Legal NERC with ontologies, Wikipedia and curriculum learning

Cristian Cardellino, Milagro Teruel, Laura Alonso Alemany, Serena Villata

▶ **To cite this version:**

Cristian Cardellino, Milagro Teruel, Laura Alonso Alemany, Serena Villata. Legal NERC with ontologies, Wikipedia and curriculum learning. 15th European Chapter of the Association for Computational Linguistics (EACL 2017), 2017, Valencia, Spain. pp.254 - 259, 10.18653/v1/E17-2041 . hal-01572444

HAL Id: hal-01572444
https://hal.science/hal-01572444

Submitted on 7 Aug 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Legal NERC with ontologies, Wikipedia and curriculum learning*

Cristian Cardellino[1], Milagro Teruel[1], Laura Alonso Alemany[1] and Serena Villata[2]

[1]Natural Language Processing Group, FaMAF-UNC, Crdoba, Argentina
[2]Université Côte d'Azur, CNRS, Inria, I3S, France
[1]{crscardellino,mteruel}@gmail.com,alemany@famaf.unc.edu.ar
[2]villata@i3s.unice.fr

Abstract

In this paper, we present a Wikipedia-based approach to develop resources for the legal domain. We establish a mapping between a legal domain ontology, LKIF (Hoekstra et al., 2007), and a Wikipedia-based ontology, YAGO (Suchanek et al., 2007), and through that we populate LKIF. Moreover, we use the mentions of those entities in Wikipedia text to train a specific Named Entity Recognizer and Classifier. We find that this classifier works well in the Wikipedia, but, as could be expected, performance decreases in a corpus of judgments of the European Court of Human Rights. However, this tool will be used as a preprocess for human annotation.

We resort to a technique called *curriculum learning* aimed to overcome problems of overfitting by learning increasingly more complex concepts. However, we find that in this particular setting, the method works best by learning from most specific to most general concepts, not the other way round.

1 Introduction

Many legal ontologies have been proposed in the literature with different purposes and applied to different sub-domains, e.g., (Ajani et al., 2009; Hoekstra et al., 2007; Athan et al., 2015). However, their manual creation and maintenance is a very time-consuming and challenging task: domain-specific information needs to be created by legal experts to ensure the semantics of regulations is fully captured. Such ontologies have little coverage, because they have a small number of entities or dwell only in concepts, not concrete entities. Moreover, only very few annotated legal corpora exist where entities can be gathered from. All this constitutes an important barrier for Information Extraction from legal text.

There is little work on increasing the coverage of legal ontologies. Bruckschen *et al.* (2010) describe a legal ontology population approach through an automatic NER to legal data. Lenci *et al.* (2009)'s ontology learning system T2K extract terms and their relations from Italian legal texts, and it is able to identify the classes of the ontology. Humphreys *et al.* (2015) extract norm elements (norms, reasons, powers, obligations) from European Directives using dependency parsing and semantic role labeling, taking advantage of the structured format of the Eunomos legal system. Boella *et al.* (2014) exploit POS tags and syntactic relations to classify textual instances as legal concepts. All these approaches rely on an important amount of domain knowledge and hand-crafted heuristics to delimit legal concepts and how they are expressed in text.

In contrast, we take an unexpensive approach, exploiting the information already available in Wikipedia and connecting it with ontologies. We establish a mapping between the WordNet- and Wikipedia-based YAGO ontology [1] and the LKIF ontology[2] for the legal domain. By doing this, we are transferring the semantics of LKIF to Wikipedia entities and populating the LKIF ontology with Wikipedia entities and their mentions.

However, even using Wikipedia, many of the classes have few instances. To adress the problems of training with few instances, we apply a learning strategy called *curriculum learning* (Bengio et al., 2009). Roughly, curriculum learning is a method that trains a model incrementally, by presenting

*The authors have received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skodowska-Curie grant agreement No 690974 for the project MIREL: MIning and REasoning with Legal texts.

[1]www.yago-knowledge.org/

[2]http://www.estrellaproject.org/lkif-core/

to it increasingly more complex concepts. This should allow to find the most adequate generalizations and avoid overfitting. However, we find that curriculum learning does not produce the expected improvements. On the contrary, *reversed* curriculum learning, learning from most specific to most general, produces better results, and it helps to indicate that there may be incoherences in the mappings between ontologies.

2 Exploiting Wikipedia to populate an ontology of the legal domain

Wikipedia has been used as a corpus for NERC because it provides a fair amount of naturally occurring text where entities are tagged and linked to an ontology, i.e., the DBpedia (Hahm et al., 2014) ontology. One of the shortcomings of such approach is that not all entity mentions are tagged, but it is a starting point to learn a first version of a NERC tagger, which can then be used to tag further corpora and alleviate the human annotation task.

2.1 Domain and classes to be learned

Our target domain is formally represented by the well known LKIF ontology (Hoekstra et al., 2007), which provides a model for core legal concepts. In order to transfer the semantics of LKIF to the relevant annotated entities in Wikipedia, we manually define a mapping between the extended LKIF[3] and YAGO (Suchanek et al., 2007), a Wikipedia-based principled ontology.

We do not map relations but only classes. The mapping is from a node in one ontology to another node in the other ontology. All children nodes of a connected node are connected by their most immediate parent. Therefore, all children nodes of the mapped YAGO nodes are effectively connected to LKIF through this mapping.

There are a total of 69 classes in this portion of the LKIF ontology, of which 30 could be mapped to a YAGO node, either as children or as equivalent classes. Two YAGO classes were mapped as parent of an LKIF class, although these we are not exploiting in this approach.

From YAGO, 47 classes were mapped to a LKIF class, with a total of 358 classes considering their children, and summing up 4'5 million mentions.

[3]The extended LKIF covers the classes in `norm.owl`, `legal-action.owl` and `legal-role.owl`, which covers core concepts of the legal domain, and not in the rest of the LKIF ontology, which provides auxiliary concepts.

Level 2 NERC (6 classes, all populated)	**Level 3** LKIF (69 classes, 21 populated)	**Level 4** YAGO (358 classes, 122 populated)
Person	Legal Role ...	judge lawyer ...
Organization	Company Corporation Public Body ...	company limited company corporation foundation court ...
Document	Regulation Contract ...	legal code law contract ...
Abstraction	Legal Doctrine Right ...	case law liberty indebtedness ...
Act	Statute ...	legislative act ..

Figure 1: Levels of abstraction of our ontology.

Curriculum learning requires that concepts are organized in a hierarchy. We did not use the hierarchy provided by the two ontologies themselves because LKIF is not hierarchical, but more aimed to represent interrelations and mereology. That is why we developed a hierarchy of concepts displayed in Figure 1. The top distinction is between Named Entities and non-Named Entities, then within Named Entities we distinguish Person, Organization, Document, Abstraction and Act, within those we distinguish LKIF classes and within those we distinguish YAGO classes.

2.2 Training corpus

To build our corpus, we considered as tagged entities the spans of text that are an anchor for a hyperlink whose URI is one of the mapped entities. Then, we extracted sentences that contained at least one named entity.

Then, words within the anchor span belong to the I class (**I**nside), outside the span, to the O class. The O class made more than 90% of the instances, so we randomly subsampled non-named entity words to make it at most 50% of the corpus, so that classifiers would not be too biased. Thus

built, the corpus consists of 21 million words.

The corpus was divided into three parts: 80% of the corpus for training, 10% for tuning and 10% for testing. The elements on each part were randomly selected to preserve the proportion of each class in the original corpus, with a minimum of one instance of each class appearing in each part. We consider only entities with a Wikipedia page and with more than 3 mentions in Wikipedia.

3 NERC with Curriculum Learning

Curriculum learning (CL) is a continuation method (Allgower and Georg, 2012), i.e. an optimization strategy for dealing with minimizing non-convex criteria, like neural networks classifiers. The basic idea of this method is to first optimize a smoothed objective, in our case, more general concepts, and then gradually consider less smoothing, in our case, more specific concepts. The underlying intuition is that this approach reveals the global picture (Bengio et al., 2009).

We applied curriculum learning with the following rationale. First, a neural network with randomly set weights is trained to distinguish NE vs. non-NE. Once this classifier has converged, the weights obtained are used as the starting point of a classifier with a similar architecture (in number of layers and number of neurons per layer), but with more specific classes. In our case, the classification divides the examples in the six classes Person, Organization, Document, Abstraction, Act, non-NE. Again when this classifier converges, its weights are used for the next level of classification, the LKIF concepts, and finally the YAGO classes.

Let us consider the following example: we start with the text "Treaty of Rome", then in the first iteration we train the classifier to learn it as a *NE*; the second iteration classifies it as a *Document*; in the third iteration it falls in the LKIF *Treaty* class, and finally, in the last iteration, it is linked to the YAGO *wordnet_treaty_106773434*.

When we trained the neural network, We carried out experiments with one, two and three hidden layers, but a single hidden layer, smaller than the input layer, performed better, so we set this as the architecture for neural networks. In each iteration of CL only the output layer is modified to suit the abstraction of the classes to the corresponding step of the CL iteration, leaving the hidden layer and the weights from the input to the hidden layer.

3.1 Representation of examples

We represented examples with a subset of the features proposed by (Finkel et al., 2005) for the Stanford Parser CRF-model. For each instance (each word) we used: current word, current word PoS-tag, all the n-grams ($1 <= n <= 6$) of characters forming the prefixes and suffixes of the word, the previous and next word, the bag of words (up to 4) at left and right, the tags of the surrounding sequence with a symmetric window of 2 words and the occurrence of a word in a full or part of a gazetteer. The final vector characterizing each instance had more than 1.5e6 features, too large to be handled due to memory limitations. In addition, the matrix was largely sparse. As a solution, we applied a simple feature selection technique using Variance Threshold. We filtered out all features with variance less than 2e-4, reducing the amount of features to 10854.

4 Evaluation

We evaluated a neural network classifier comparing batch learning and curriculum learning. As a comparison ground, we also trained a linear classifier, namely a Support Vector Machine (SVM) with a linear kernel, and the Stanford CRF Classifier model for NER (Stanford NLP Group, 2016), training it with our corpus with Wikipedia annotations for the LKIF classes. For the Stanford NERC, we use the same features as the MLP classifiers, except the features of presence in gazetteers and the PoS tags of surrounding words. Decision trees and Naive Bayes (NB) classifiers were discarded because the cardinality of the classes was too large for those methods.

To evaluate the performance, we computed accuracy, precision and recall in a word-to-word basis in the test portion of our corpus. For this particular problem, the performance for the majority class, *non-NE*, eclipses the performance in the rest. To have a better insight on the performance, we also provide macro-average of precision and recall *without the non-NE class*. Macro-average is showing differences in all classes, with less populated classes comparable to more populated ones.

The difference in performance between different classifiers was very small. To assess the statistical significance of results, we applied a Student's t-test with paired samples comparing classifiers. We divided the Wikipedia corpus in five disjunct subcorpora, then divided those

in train/validation/test, compared results and obtained p-values for the comparison.

4.1 Performance in a legal corpus

In order to evaluate the performance of this approach in legal corpora like norms or case-law, we manually annotated a corpus of judgments of the European Court of Human Rights, identifying NEs that belong to classes in our ontology or to comparable classes that might be added to the ontology. We annotated excerpts from 5 judgments of the ECHR, totalling 19,000 words. We identified 1,500 entities, totalling 3,650 words. Annotators followed specific guidelines, inspired in the LDC guidelines for annotation of NEs (Linguistic Data Consortium, 2014).

There were 4 different annotators. The agreement between judges ranged from $\kappa = .4$ to $\kappa = .61$, without significant differences across levels of granularity. Most of the disagreement between annotators was found for the recognition of NEs, not for their classification. The inter-annotator agreement obtained for this annotation is not high, and does not guarantee reproducible results, but it is useful for a first assessment of performance.

5 Analysis of results

The results on the test portion of our Wikipedia corpus are reported in Table 1. We show overall accuracy, and the average recall and precision across classes other than the non-NE class. It can be seen that neural network classifiers perform better than both SVM and the Stanford NER. Differences are noticeable when the non-NE class is not included in the metric, as in the non-weighted average of precision and recall without non-NEs.

It can be observed that curriculum learning does not introduce an improvement in accuracy over batch learning in a neural network. As explained in the previous Section, we applied the paired t-test in five different samples of the corpus to assess whether the difference between classifiers was significant or not, and we found that two out of five of the obtained results were not significantly different ($p < 0.05$), but the other three were. Therefore, it seems that Curriculum Learning, at least the way we applied it here, does not introduce an improvement.

We further analyzed the results and we found that the MLP classifier performs far better in smaller classes (with less instances) than in bigger classes, for all levels of abstraction but most dramatically for the LKIF level, where F-score for the 20% biggest classes drops to .11 (in contrast with .62 for NERC and .42 for YAGO), while for the smallest classes it keeps within the smooth decrease of performance that can be expected from the increase in the number of classes, and thus an increase in the difficulty of classification.

	accuracy	precision	recall	F1
NER (2 classes)				
SVM	1.00	.54	.06	.11
Stanford	.88	.87	.87	.87
MLP	1.00	**1.00**	**1.00**	1.00
NERC (6 classes)				
SVM	.97	.37	.18	.24
Stanford	.88	.78	.82	.79
MLP	.99	.89	**.83**	.85
CL	.99	**.91**	.81	.85
LKIF (21 classes)				
SVM	.93	.53	.26	.35
Stanford	.97	**.84**	**.71**	**.77**
MLP	.97	.73	.65	.68
CL	.97	.71	.62	.66
YAGO (122 classes)				
SVM	.89	.51	.25	.34
MLP	.95	.76	.64	**.69**
CL	.95	**.77**	.64	.68

Table 1: Results for the test portion of the Wikipedia corpus. Accuracy figures consider non-NEs, but precision and recall are an average of all classes (macro-average) except the majority class of non-NEs. The results for the NER level for Curriculum Learning are the same as for MLP, and the Stanford NER cannot handle the number of classes in the YAGO level.

These results corroborate an observation that has already been anticipated in general results, namely, that the LKIF level of generalization is not adequate for automated NERC learnt from the Wikipedia, because the NERC cannot distinguish the classes defined at that level, that is, in the original LKIF ontology. In contrast, the NERC does a better job at distinguishing YAGO classes, which are natively built from Wikipedia, even if the classification problem is more difficult because of the bigger number of classes.

On the other hand, the fact that smaller classes are recognized better than bigger classes indicates that bigger classes are ill-delimited. It may be that

these classes are built as catch-all classes, grouping heterogeneous subclasses. That indicates that curriculum learning might work better learning first from most concrete classes, then from more general classes. In Table 2 we show the performance of curriculum learning in reverse, that is, from the smallest classes to the most general ones.

	accuracy	precision	recall	F1
NER (2 classes)				
MLP	1.00	1.00	1.00	1.00
rev CL	1.00	1.00	1.00	1.00
NERC (6 classes)				
MLP	.99	.89	.83	.85
CL	.99	.91	.81	.85
rev CL	.99	**.93**	.83	**.87**
LKIF (21 classes)				
MLP	.97	**.73**	**.65**	**.68**
CL	.97	.71	.62	.66
rev CL	.97	.70	.62	.65
YAGO (122 classes)				
MLP	.95	.76	.64	**.69**
CL	.95	**.77**	.64	.68

Table 2: Comparison of curriculum learning strategies, from most general to most specific (CL) and from most specific to most general (rev CL), with accuracy including the class of non-NEs and macro-average excluding the class of non-NEs.

It can be seen that curriculum learning from most specific to most general provides the best result for the NERC level of abstraction, outperforming the other two neural approaches. However, at the LKIF level, the batch approach performs better. This seems to indicate that, for this particular hierarchy and dataset, curriculum learning seems more adequately applied from most specific to most general. Moreover, the YAGO and NERC levels seem to be coherent with each other, while the LKIF level seems disconnected from the other two.

Therefore, it seems that the chosen level of granularity for legal NERC using our ontology should be either the 6-class level or the YAGO level, depending on the level of granularity that is required. Moreover, the mapping between YAGO and LKIF needs to be further consolidated.

5.1 Performance in a legal corpus

The results for different approaches to NERC trained on Wikipedia, with the corpus of judgments of the ECHR described in Section 4.1 are shown in Table 3. We can see that the drop in performance with respect to results on Wikipedia is very important, but on the other hand this annotator has no annotation cost, because examples are obtained from Wikipedia, so it can be considered as a preprocess for human validation / annotation of legal text.

	accuracy	precision	recall	F1
NER (2 classes)				
Stanford	.78	.60	.35	.33
MLP	.54	.76	.55	.43
NERC (6 classes)				
Stanford	.75	.38	.19	.19
MLP	.53	.64	.25	.25
LKIF (21 classes)				
Stanford	.78	.09	.05	.05
MLP	.77	.35	.15	.17
YAGO (122 classes)				
MLP	.89	.16	.08	.08

Table 3: Comparison of different strategies for NERC trained on Wikipedia, as they perform in ECHR judgments.

6 Conclusions and Future Work

We have presented an approach to ontology population in the legal domain by exploiting annotations from Wikipedia, and mapping them to the legal ontology LKIF via the YAGO ontology. We have aligned the LKIF and YAGO ontologies, we have obtained a Named Entity Recognizer and Classifier for the legal domain, and we have populated the LKIF ontology at the same time.

We have shown that the machine learning technique of curriculum learning produces slightly (but significantly) better classifiers than the same classifier with batch learning, but only if applied from most specific to most general classes, and only in levels of generality that are coherent with each other.

Further work will be aimed to a more insightful error analysis with the aim to guide a better mapping between YAGO and LKIF. We will also enhance and consolidate the annotation of judgments from the European Court on Human Rights using these classifiers as pre-annotation, in combination with the Stanford NERC, and resorting to Active Learning techniques.

References

Gianmaria Ajani, Guido Boella, Leonardo Lesmo, Marco Martin, Alessandro Mazzei, Daniele P. Radicioni, and Piercarlo Rossi. 2009. Legal taxonomy syllabus version 2.0. *IDT*, page 9.

Eugene L. Allgower and Kurt Georg. 2012. *Numerical continuation methods: an introduction*, volume 13. Springer Science & Business Media.

Tara Athan, Guido Governatori, Monica Palmirani, Adrian Paschke, and Adam Wyner. 2015. LegalRuleML: Design principles and foundations. In Wolfgang Faber and Adrian Pashke, editor, *The 11th Reasoning Web Summer School*, pages 151–188, Berlin, Germany, jul. Springer.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 41–48, New York, NY, USA. ACM.

Guido Boella, Luigi Di Caro, Alice Ruggeri, and Livio Robaldo. 2014. Learning from syntax generalizations for automatic semantic annotation. *J. Intell. Inf. Syst.*, 43(2):231–246.

Mrian Bruckschen, Caio Northfleet, Douglas da Silva, Paulo Bridi, Roger Granada, Renata Vieira, Prasad Rao, and Tomas Sander. 2010. Named entity recognition in the legal domain for ontology population. In *3rd Workshop on Semantic Processing of Legal Texts (SPLeT 2010)*.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 363–370, Stroudsburg, PA, USA. Association for Computational Linguistics.

Younggyun Hahm, Jungyeul Park, Kyungtae Lim, Youngsik Kim, Dosam Hwang, and Key-Sun Choi. 2014. Named entity corpus construction using wikipedia and dbpedia ontology. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Hrafn Loftsson, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland, may. European Language Resources Association (ELRA).

R. Hoekstra, J. Breuker, M. Di Bello, and A. Boer. 2007. The lkif core ontology of basic legal concepts. In *Proceedings of the Workshop on Legal Ontologies and Artificial Intelligence Techniques (LOAIT 2007)*.

Llio Humphreys, Guido Boella, Livio Robaldo, Luigi di Caro, Loredana Cupi, Sepideh Ghanavati, Robert Muthuri, and Leendert van der Torre. 2015. Classifying and extracting elements of norms for ontology population using semantic role labelling. In *Proceedings of the Workshop on Automated Detection, Extraction and Analysis of Semantic Information in Legal Texts*.

A. Lenci, S. Montemagni, V. Pirrelli, and G. Venturi. 2009. Ontology learning from italian legal texts. In *Proceeding of the 2009 Conference on Law, ontologies and the Semantic Web: Channelling the Legal information Flood*.

Linguistic Data Consortium. 2014. Deft ere annotation guidelines: Entities v1.7. http://nlp.cs.rpi.edu/kbp/2014/ereentity.pdf.

Stanford NLP Group. 2016. Stanford named entity recognizer (ner). http://nlp.stanford.edu/software/CRF-NER.shtml.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 697–706, New York, NY, USA. ACM.