Learning Slowly To Learn Better: Curriculum Learning for Legal Ontology Population

Cristian Cardellino, Milagro Teruel, Laura Alonso Alemany
Natural Language Processing Group
FaMAF-UNC, Cordoba, Argentina
{cardellino,teruel,alemany}@famaf.unc.edu.ar

Serena Villata
Université Côte d'Azur
CNRS, Inria, I3S, France
villata@i3s.unice.fr

Abstract

In this paper, we present an ontology population approach for legal ontologies. We exploit Wikipedia as a source of manually annotated examples of legal entities. We align YAGO, a Wikipedia-based ontology, and LKIF, an ontology specifically designed for the legal domain. Through this alignment, we can effectively populate the LKIF ontology, with the aim to obtain examples to train a Named Entity Recognizer and Classifier to be used for finding and classifying entities in legal texts. Since examples of annotated data in the legal domain are very few, we apply a machine learning strategy called *curriculum learning* aimed to overcome problems of overfitting by learning increasingly more complex concepts. We compare the performance of this method to identify Named Entities with respect to batch learning as well as two other baselines. Results are satisfying and foster further research in this direction.

Introduction

Ontologies are the main mechanism for domain-specific knowledge representation as they allow for an exhaustive characterization of such domain. A special class of ontologies are the legal ones which specify legal concepts in a formal way, such that reasoning mechanisms can then be exploited over such information (Sartor et al. 2013). Many legal ontologies have been proposed in the literature with different purposes, e.g., (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 compliance with the regulations we aim at modeling and capture their full semantics. First of all, legal ontologies need to specify carefully the legal concepts highlighting possible conflicts among them and further subtle issues specific of the legal domain, and second, such ontologies have little coverage, i.e., they have a small number of entities and only very few annotated legal corpora exist where entities can be gathered from. This lack of coverage of legal ontologies makes it difficult to train legal Named Entity Recognition (NER) systems, and to support Entity Linking to mine legal documents.

In this paper, we present an approach to legal ontology population with the aim to find new entities to ease the next Entity Linking step. We apply a learning strategy, called *curriculum learning* (Bengio et al. 2009), that can deal with the lack of examples that characterizes such kind of ontologies, but that still produces few errors only. Roughly, curriculum learning is a method that trains a model by presenting increasingly more complex concepts. In our approach, we implement curriculum learning to learn progressively finer grained classes using a neural network classifier.

We choose Wikipedia, a source of manually annotated examples of legal entities, to create our legal domain Named Entity Recognizer and Linker. However, this choice presents the following caveats: *(i)* Wikipedia-linked ontologies (e.g., YAGO – Yet Another Great Ontology) are not specifically targeted to represent the legal domain, *(ii)* Wikipedia as a source of annotated examples has the peculiarity that not all instances of a given entity are tagged as such, although all that are tagged as such can be assumed to be tagged correctly, and *(iii)* Wikipedia does not belong to any genre of legal document, even if some of its articles dwell in the legal domain. To address these shortcomings, we align the the YAGO ontology[1], i.e., a huge semantic knowledge base, derived from Wikipedia, WordNet and GeoNames, with the LKIF (Legal Knowledge Interchange Format) ontology [2], an ontology specifically designed for modeling legal knowledge. In this way, we are not only transferring the semantics of LKIF to Wikipedia entities, but also populating the *assertion components* (i.e., A-Box) of the LKIF ontology with Wikipedia entities and their mentions.

We compare the performance of curriculum learning in the Named Entity Recognition and Classification (NERC) task with respect to those obtained by applying batch learning, a method which generates the best predictor by learning on the entire training data set at once. Results are promising, and foster research in this direction.

The rest of the paper is as follows. First, we discuss the related literature with respect to the proposed approach, and second, we describe our approach to ontology population via curriculum learning. Finally, the evaluation is reported, together with the error analysis.

[1] http://bit.ly/1mgcWhJ
[2] http://www.estrellaproject.org/lkif-core/

Related work

The task of Named Entity Recognition and Linking has gained a lot of popularity since the '90es, and several benchmark datasets to test, adapt, and improve entity recognition and linking have been proposed in the literature. The main topical domains considered to construct these benchmarks have been for instance science, sports, politics, music, catastrophic events, leading to the development of corpora extracted from news articles (Hoffart et al. 2011), tweets (Derczynski et al. 2015), blog articles (Waitelonis, Exeler, and Sack 2015), scientific articles (Finkel et al. 2004). However, up to our knowledge, no corpus has been built to test, adapt, and improve entity recognition and linking over legal texts.

In the literature, only few approaches addressed the problem of legal ontology population. More precisely, (Bruckschen et al. 2010) describes an ontology population approach to legal data, whose experimental evaluation is run over a corpus of legal and normative documents for privacy. Ontology population is then obtained through the task of NER. (Lenci et al. 2009) report an experiment on an ontology learning system called T2K. They use NLP and Machine Learning methods to extract terms and relations from free text. The experimental evaluation is conducted on Italian legal texts, and it is able to identify the classes of the ontology, as well as many hyponymy relations. Related approaches are presented by (Humphreys et al. 2015) and (Boella et al. 2014). The former discusses the results of the classification and extraction task of norm elements in European Directives using dependency parsing and semantic role labeling. This approach focuses on how to extract prescriptions (i.e., norms) and other concepts (e.g., reason, power, obligation, nested norms) from legislation, and how to automate ontology construction. Similarly, (Boella et al. 2014) propose an approach that provides POS tags and syntactic relations as input of a SVM to classify textual instances to be associated to legal concepts.

The main difference with all the above mentioned approaches is the generality of the approach we propose in this paper, that can be easily adapted to any legal ontology and that shows good performance via curriculum learning. Moreover, the goal of our approach, i.e., Named Entity Recognition and Entity Linking, and the populated ontologies respectively, are different.

Ontology population via curriculum learning

One of the main issues in learning Named Entity Recognition and Classification with very few examples is that of data sparseness (i.e., small coverage and lack of generalization), normally expressed as overfitting models. Ontologies are a valuable resource to overcome this limitation. They represent concepts organized in different ways, and one of them in particular, the *is-a* relation, can be considered as a representation of abstraction. Therefore, we can exploit this relation to introduce abstraction and avoid overfitting.

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 network classifiers. The basic idea of this method is to first optimize a smoothed objective, and then gradually consider less smoothing, with the intuition that "a smooth version of the problem reveals the global picture" (Bengio et al. 2009). This method allows us to take advantage of the abstractions provided by ontologies, as more general categories serve as smoothed objective, to guide the learning process into more concrete concepts, serving as less smoothed objectives.

Wikipedia has been used as a corpus for NERC because it provides a fair amount of naturally occurring text where (part of the) 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 in Wikipedia, 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.

Ontology alignment

Our target domain is formally represented by the well-known LKIF ontology (Hoekstra et al. 2007), which provides a model for core legal concepts. The LKIF core legal ontology consists of 15 modules, each of which describes a set of closely related concepts from both legal and commonsense domains. In order to align the semantics of LKIF to the relevant annotated entities in Wikipedia, we manually define a matching from the three modules that form the LKIF legal ontology, namely *legal action* (i.e., a number of legal concepts related to action and agent, such as public acts, public bodies, legal person), *legal role* (i.e., a small number of legal concepts related to roles, legal professions) and *norm* (i.e., the expression module where norms are defined as qualifications), and YAGO (Suchanek, Kasneci, and Weikum 2007), a Wikipedia-based principled ontology.

We do not match relations but only classes. The alignment 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 aligned YAGO nodes are effectively connected to LKIF through this matching. There is a total of 69 classes in this portion of the LKIF ontology, of which 30 could be mapped to a YAGO node. Additionally, 9 YAGO nodes are mapped as children of 5 LKIF classes. 55% of the classes of LKIF could not be mapped to a YAGO node, because they were too abstract (i.e., *Normatively_Qualified*), there was no corresponding YAGO node circumscribed to the legal domain (i.e., *Mandate*), there was no specific YAGO node (i.e., *Mandatory_Precedent*) or the YAGO concept was overlapping but not roughly equivalent (as for "*agreement*" or "*liability*")[3].

Training corpus

To build our corpus, we downloaded a `XML` dump of the English Wikipedia[4] from March 2016, and we processed it via the WikiExtractor (of Pisa 2015) to remove all the `XML` tags and Wikipedia markdown tags, but leaving the links. We extracted all those articles that contained a link to an entity

[3]The alignment is available at `http://bit.ly/2lqPgdr`.

[4]`https://dumps.wikimedia.org/`

of YAGO that we aligned to LKIF. We considered as tagged entities only the spans of text that are an anchor for a hyperlink whose URI is one of the mapped entities. We obtained a total of 4,5 million mentions, spanning a total of 10 million words, corresponding to 102,000 unique entities. Then, we extracted sentences in each document that contained at least one named entity.

We consider the problem of NERC as a word-based representation, i.e., each word represents a training instance. Words within the anchor span belong to the I class (**I**nside a Named Entity), others to the O class (**O**utside a Named Entity). The O class made more than 90% of the instances. This imbalance in the classes results in largely biased classifiers, so we randomly subsampled non-named entity words to make it at most 50% of the corpus.

The corpus was divided into three parts: 80% of the corpus was selected for training, 10% was selected for testing, and finally 10% was left for validation. 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.

Features

We represented examples with a subset of the features proposed by (Finkel, Grenager, and Manning 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.

Experimental setting

Curriculum learning is based on the idea of learning gradually by taking the model learnt for simpler (abstract) concepts as starting point for a model with more complex (concrete) concepts, using a Multilayer Perceptron (MLP).

As a comparison ground, we also trained a linear classifier, namely a Support Vector Machine (SVM) with a linear kernel, and the Stanford CRFClassifier model for NER. Decision trees and Naive Bayes (NB) classifiers were discarded because the cardinality of the classes was too large for those methods to handle.

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 person/non-person/non-NE. Again when this classifier converges, its weights are used for the next level of classification, advancing until reaching the last step where classes are the most specific ones.

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 *non-person*; in the third iteration it falls in the *Treaty* class, and finally, in the last iteration, it is linked to the URI *Treaty_of_Rome*.

In this paper, we use 4 levels of generalization of classes:

1. Named Entity / non Named Entity
2. Named Entities Person / Non-Person
3. Named Entities LKIF Classification (NEL): this iteration classifies into classes of the LKIF ontology, comparable to classical Named Entity Classification.
4. Named Entities URIs Classification (NEU): Each mention of a Named Entity is classified as its most specific YAGO node, comparable to Entity Linking.

These four levels are not based on the structure of the underlying ontologies, but they represent a coarse-grained stratification of the concepts by their generality.

Figure 1 shows the two neural networks architectures we implemented. In a first approach to curriculum learning, we take a neural network, and each iteration of CL only modifies the output layer to suit the abstraction of the classes to the corresponding step of the CL iteration, leaving the hidden layers exactly the same. For the first iteration of CL (NER), the output layer has two neurons (NE and non-NE), the next CL iteration replaces it with a new output layer with 3 neurons (Person, Non-Person and Non-NE), and so on, replacing the output layer according to the classification task (NEL, NEU).All the layers prior to the last one are

Our second approach to curriculum learning consists of gradually removing extra layers of the network. The first iteration (NER) has a hidden layer with a number of neurons equal to the number of classes of YAGO URIs, the following layer has as many neurons as the number of classes of the NEL iteration, the following layer represents the classes of the NEP iteration and the output layer has the two neurons of the NER iteration. When the classifier converges the NER layer is removed and the NEP layer is used as output layer for the next iteration, and so on until the last iteration, that uses the biggest and innest layer, NEU, as the output layer.

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 decided for this configuration.

Evaluation

In order to assess the goodness of the Curriculum Learning approach, we evaluated the accuracy in the test portion of the corpus of different approaches with respect to the tasks of NERC (classifying entity mentions into LKIF classes) and NE Linking (classifying mentions into YAGO URIs).

We evaluated a neural network classifier with the configuration described in the previous section, and we considered three settings: batch learning, curriculum learning by changing the output layer, and curriculum learning by removing layers. We also evaluated a Support Vector Machine

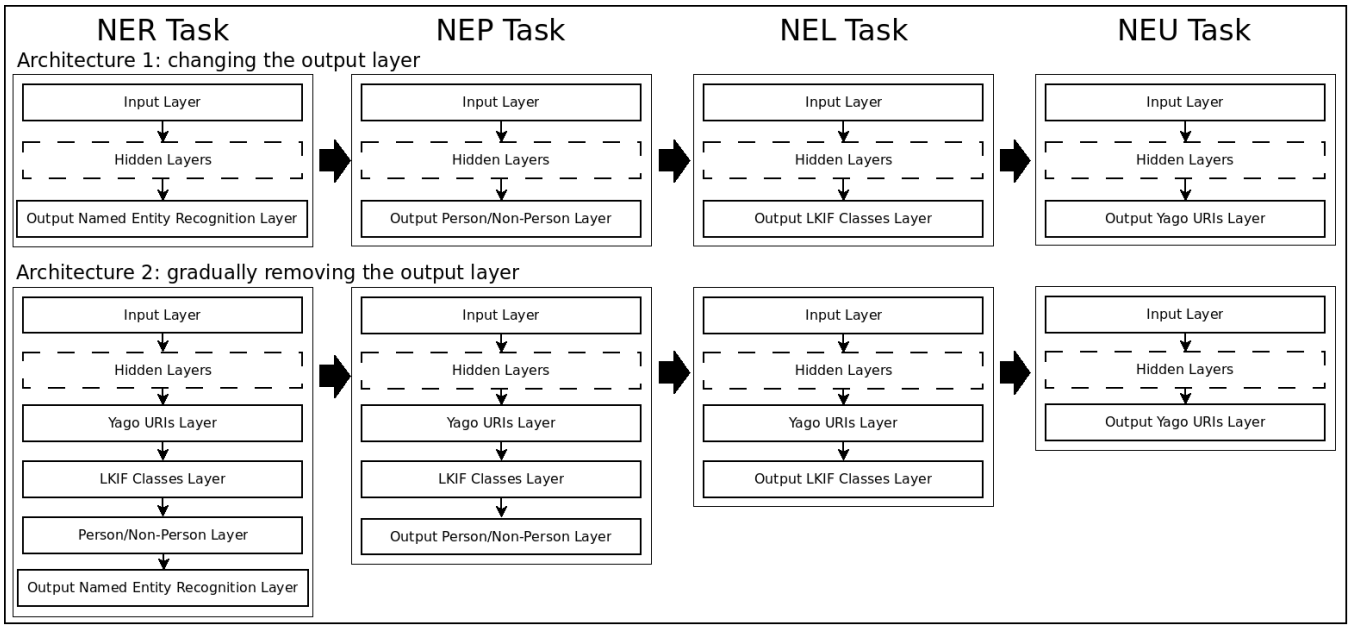

Figure 1: Architectures of neural networks used in Curriculum Learning.

to compare a linear classifier against the neural network approach. As an additional baseline, we obtained the performance of the Stanford NER system (Stanford NLP Group 2016), training it with our corpus with Wikipedia annotations for the LKIF classes. We did not apply it for Entity Linking to YAGO URIs because this tool is not intended for Entity Linking. 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.

To evaluate the performance, we computed accuracy, precision and recall in a word-to-word basis in the test portion of our corpus. That is, we calculated if each word has been correctly or incorrectly tagged. For precision and recall, we calculated the weighted average e of the precision and recall for each class excluding the non-Named Entity class, to get a better insight of performance.

It is worth noticing that this evaluation is partial because not all entity mentions are tagged in our test corpus. Indeed, since this test corpus is part of Wikipedia, there are many mentions that are not effectively tagged in the corpus. To effectively assess recall, we manually tagged 10 randomly selected Wikipedia articles for a total of 6,000 words. In those articles, we tagged all mentions of Named Entities in the LKIF domain, and evaluated the performance of our approaches for this corpus. This evaluation also provides insights for the use of automatic classifications as input for manual annotation.

Analysis of results

The results on the test portion of our Wikipedia corpus are reported in Table 1 for Named Entity Recognition and Classification with LKIF classes, and in Table 2 for Named Entity Linking to YAGO URIs. We show overall accuracy, and the mean recall and precision across classes other than the O class (no Named Entity).

It can be seen that neural network classifiers perform better than both SVM and the Stanford NER. Differences are more noticeable in precision and recall figures, where the good performance in the O class does not obscure the results. It can also be observed that curriculum learning does introduce an improvement in accuracy over batch learning in a neural network, although very slight. To assess the statistical significance of this slight difference, we applied a Student's t-test with paired samples comparing all pairs of classifiers for both approaches. We divided the Wikipedia corpus in five disjunct sub-corpora, then divided those in train/validation/test, and compared the results of the neural network experiments in each of them. In average, two out of five of the obtained results were not significant ($p < 0.05$). Therefore, the different neural network approaches were not very different, but still some differences arise. For the purpose of using these classifiers as a starting point for manual annotation of legal texts, even a slight improvement is useful, as it implies an important reduction in annotation cost.

Even with such a performance in the CL learning and batch methods, the CL approach of changing the last layer of the MLP obtains better precision than the batch learning method, and better recall than both alternatives. Recall is most important for our purposes of using this classifier as a starting point for manual annotation.

Results for LKIF classes classification

If we examine the results for NERC with LKIF classes, we can see that all the neural network approaches have a much higher performance than the two baseline methods, with a relative improvement of 7% in precision and 11% in recall.

approach	acc	prec	rec	fscore
Baselines				
SVM	.971	.950	.777	.855
Stanford NER	.982	.909	.866	.886
MLP	.991	.972	.961	.966
Curriculum Learning				
change layer	**.992**	**.978**	**.965**	**.971**
remove layer	.990	.978	.950	.964

Table 1: Comparison of the results of different approaches in assigning entity mentions to their correct LKIF class.

If we examine the confusion matrix[5], we can see that the CL classifier that removes layers presents more errors in the less populated classes, that are confused with the other classes (a total of 9% against 5% for the other two approaches). Also, the CL classifier that changes layers is slightly better than the neural network without CL at the "most" populated classes. This higher recall in classes other than O (see Table 1) is useful to help humans label NEs.

The accuracy of the Support Vector Machine classifier and the Stanford CRF classifier are visually lower than any neural network approach. Particularly, we see a clear pattern where most of misclassified classes are mostly classified as Non NE for the Stanford system. This does not happen with the neural networks, where non NE are distinguished from NE and misclassification occurs within the NE, most likely a result of better internal representations the neural networks have.

Results for YAGO Entity Linking

approach	acc	prec	rec	fscore
Baselines				
SVM	.897	.451	.342	.389
MLP	.932	.471	.446	.458
Curriculum Learning				
change layer	.933	**.475**	**.450**	**.462**
remove layer	.933	.472	.448	.460

Table 2: Comparison of the results of different approaches in assigning entity mentions to their correct YAGO URI.

The task of assigning each word its correct YAGO node is more difficult than the task of assigning LKIF classes, because there are more than 10,000 classes for this task, in contrast with the potential 35 classes in LKIF. As can be expected, results are worse. Whereas the overall accuracy figures do not drop because of the dominance of the O class (which represents the non entities), average precision and recall are much lower. Stanford NER could not be applied to this task because it cannot handle that number of classes.

Neural network classifiers perform far better than the SVM baseline, but different approaches, with and without Curriculum Learning are fairly indistinguishable. Table 2 shows slightly better results for the CL approach changing the last layer in precision, but we can also see that the CL approach that removes layers is slightly better than the other two approaches in the less populated classes. We cannot conclude that one approach is clearly better than another.

It is worth noticing that the high accuracy figures and the low precision and recall figures are due to the fact that the majority class is very dominant and high accuracy in that class obscures low accuracy in the rest. Moreover, the provided precision and recall figures are an average of the figures for each class, so classes with lower figures throw down the average for all classes.

Results for the manually annotated testbed

As said before, in the Wikipedia not all mentions of an entity are annotated. Our classifiers were trained and evaluated with this partially annotated corpus, and therefore both performance and evaluation are bound to be partial. That is why we annotated ten randomly selected Wikipedia articles with at least one outgoing link to an entity mapped to a LKIF class, where we manually tagged all mentions of entities.

We evaluated the performance of classifiers in this corpus and found that overall accuracy was comparable to evaluation in the Wikipedia corpus, even if somehow smaller: around .92 for all approaches. However, we observed that 98% of errors were words that had been tagged as non-entities when they were entities. This is to be expected as most mentions of entities in Wikipedia are not tagged, and therefore automatic classifiers cannot learn their pattern of behavior. For example, in the following excerpt of the Wikipedia article on Paul K. Holmes III, the annotation provided by Wikipedia is:

```
Paul Kinloch Holmes III (born November
10, 1951) is the chief district
judge for the [United States District
Court for the Western District of
Arkansas]PublicBody .
```

Classifiers tend to recognize exactly what the Wikipedia tags as a Named Entity, but the ground truth in this case is:

```
Paul Kinloch Holmes III (born November
10, 1951) is the [chief district
judge]ProfessionalLegalRole for the [United
States District Court for the Western
District of Arkansas]PublicBody .
```

So while this example would give 100% accuracy if we compared the output of classifiers with the annotation of Wikipedia, it gives very good precision but 50% recall when compared to the specifically annotated ground truth. This confirms the intuition that we have to prioritize classifiers that have higher recall.

[5]Available here: https://dl.dropboxusercontent.com/u/15116330/CMatrix.pdf

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 LKIF ontology in the legal domain via an alignment to the YAGO ontology. We have thus obtained a Named Entity Recognizer and Classifier and Entity Linker for the legal domain and we have populated the LKIF ontology. We have evaluated classification in the LKIF classes and Entity Linking to the YAGO classes, and have obtained promising results in both. The main shortcoming of our approach is the kind of annotation available in Wikipedia, where not all mentions of Named Entities are tagged. As can be expected, the learned classifier also fails to identify many mentions of entities, as we have seen in a small manual evaluation. To address this shortcoming, we are currently manually annotating a corpus of judgments from the European Court on Human Rights using these classifiers as pre-annotation, in combination with the Stanford NERC.

We have shown that curriculum learning produces slightly better classifiers for coarse-grained NEC, but not for the Entity Linking task. This learning technique is specially suited for scenarios where classifications can be organized from most general to most specific, and learners are trained taking advantage of this organization. However, for Entity Linking we suffered from data sparseness in Wikipedia documents, so the technique did not effectively show improvements over a comparable neural network classifier without curriculum learning. However, we expect that this technique will be specially useful to progressively learn classifiers from our future annotated corpus as it is being annotated.

Acknowledgement

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.

References

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

Athan, T.; Governatori, G.; Palmirani, M.; Paschke, A.; and Wyner, A. 2015. LegalRuleML: Design principles and foundations. In *The 11th Reasoning Web Summer School*, 151–188. Springer.

Bengio, Y.; Louradour, J.; Collobert, R.; and Weston, J. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, 41–48. ACM.

Boella, G.; Caro, L. D.; Ruggeri, A.; and Robaldo, L. 2014. Learning from syntax generalizations for automatic semantic annotation. *J. Intell. Inf. Syst.* 43(2):231–246.

Bruckschen, M.; Northfleet, C.; da Silva, D.; Bridi, P.; Granada, R.; Vieira, R.; Rao, P.; and Sander, T. 2010. Named entity recognition in the legal domain for ontology population. In *3rd Workshop on Semantic Processing of Legal Texts (SPLeT 2010)*.

Derczynski, L.; Maynard, D.; Rizzo, G.; van Erp, M.; Gorrell, G.; Troncy, R.; Petrak, J.; and Bontcheva, K. 2015. Analysis of named entity recognition and linking for tweets. *Inf. Process. Manage.* 51(2):32–49.

Finkel, J.; Dingare, S.; Nguyen, H.; Nissim, M.; Manning, C.; and Sinclair, G. 2004. Exploiting context for biomedical entity recognition: from syntax to the web. In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications*.

Finkel, J. R.; Grenager, T.; and Manning, C. 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, 363–370. ACL.

Hahm, Y.; Park, J.; Lim, K.; Kim, Y.; Hwang, D.; and Choi, K.-S. 2014. Named entity corpus construction using wikipedia and dbpedia ontology. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*. ELRA.

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

Hoffart, J.; Yosef, M. A.; Bordino, I.; Fürstenau, H.; Pinkal, M.; Spaniol, M.; Taneva, B.; Thater, S.; and Weikum, G. 2011. Robust disambiguation of named entities in text. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011*, 782–792.

Humphreys, L.; Boella, G.; Robaldo, L.; di Caro, L.; Cupi, L.; Ghanavati, S.; Muthuri, R.; and van der Torre, L. 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*.

Lenci, A.; Montemagni, S.; Pirrelli, V.; and Venturi, G. 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*.

of Pisa, M. U. 2015. Wikiextractor. `http://medialab.di.unipi.it/wiki/Wikipedia_Extractor`.

Sartor, G.; Casanovas, P.; Biasiotti, M.; and Fernndez-Barrera, M. 2013. *Approaches to Legal Ontologies: Theories, Domains, Methodologies*. Springer.

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

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

Waitelonis, J.; Exeler, C.; and Sack, H. 2015. Enabled generalized vector space model to improve document retrieval. In *Proceedings of the Third NLP&DBpedia Workshop (NLP & DBpedia 2015)*, 33–44.