Relaxation Labeling for Selecting and Exploiting Efficiently Non-local Dependencies in Sequence Labeling

Guillaume Wisniewski and Patrick Gallinari

LIP6 — Université Pierre et Marie Curie
104 avenue du président Kennedy
75016 Paris France
guillaume.wisniewski@lip6.fr, patrick.gallinari@lip6.fr

Abstract. We consider the problem of sequence labeling and propose a two steps method which combines the scores of local classifiers with a relaxation labeling technique. This framework can account for sparse dynamically changing dependencies, which allows us to efficiently discover relevant non-local dependencies and exploit them. This is in contrast to existing models which incorporate only local relationships between neighboring nodes. Experimental results show that the proposed method gives promising results.

1 Introduction

Sequence labeling aims at assigning a label to each element of a sequence of observations. The sequence of labels generally presents multiple dependencies that restrict the possible labels elements can take. For example, for the task of part of speech tagging, the observations that there is only one verb in a sentence and that an article is followed either by a noun or an adjective provide valuable information to the labeling of an element. The aim of *Structured Prediction* is to develop models able to detect and exploit these dependencies so as to improve prediction performance.

Taking dependencies between labels into account entails two main difficulties: parameter estimation and complexity of inference. For the former, the more dependencies we have to consider, the more parameters we have to estimate, which creates a sparse data problem. For the latter, inferring jointly rather than independently a label sequence consistent with the dependencies often proves to be a combinatorial problem and, in the worst case, inference is known to be NP-hard to solve [1].

Several approaches have been developed for many years for sequence labeling. In order to solve the inherent difficulties of both the training and the inference steps, all current methods, like CRFs [2] or SVM'ISO [3], impose a fixed label dependency structure in which only local interactions between labels are considered: they generally incorporate only dependencies between a few successive labels. This Markov assumption limits the number of parameters to be estimated and maintains a tractable exact inference thanks to the use of dynamic programming techniques. While this assumption is critical in preserving computational

J.N. Kok et al. (Eds.): PKDD 2007, LNAI 4702, pp. 312–323, 2007.

efficiency, it is a key limitation since taking non-local dependencies into account is mandatory to achieve good performance for several tasks [4,5,6].

Two main families of approaches have been proposed to take advantage of non-local dependencies. The first one [7,8] relies on a grammar-based formalism to model non-local relationships by introducing a hierarchy of hidden variables, while the second one proposes alternative inference procedures like Gibbs sampling [9] or Integer Linear Programming [10]. In most of these methods, approximate inference algorithms are used to allow tractable inference with long-range dependencies. All these methods suffer a high complexity for both training and inference and rely on an expert knowledge to explicitly define all relevant dependencies involved.

In this paper, we propose a new approach for learning and modeling unknown dependencies among labels. Dependencies are represented using *constraints* which are logical relations among several elements and their value. This approach has several interesting properties. Firstly, it allows the dependency structure to vary according to the actual value of elements, while this structure is fixed in most existing models. Secondly, both local and long-range dependencies can be considered while preserving the computational efficiency of the training and inference steps. More precisely, following [11], we consider a two-parts model, in which, a local classifier predicts the values of variables regardless of their context, while a set of constraints maintains global consistency between local decisions. These constraints are learned and represent dependencies between labels. In this work, we use maximum entropy classifiers [12] to make local decision and relaxation labeling [13] to efficiently build an approximate solution, that satisfies as many constraints as possible.

The paper is organized as follows. We first formalize the task and explain the difficulty of incorporating non-local dependencies in representative existing approaches in Section 2. Our approach is presented in Section 3. Related work is reviewed in Section 2.3 and experimental results are presented in Section 4.

2 Background

2.1 Formalization of the Task

Sequence labeling consists in assigning a label to every element of a sequence of observations. Let $\mathbf{x} = (x_i)_{i=1}^n$ be a sequence of n observations and y_i be the label of the i^{th} element of this sequence. The sequence of labels denoted $\mathbf{y} = (y_i)_{i=1}^n$ can be seen as a *macro-label* [3] describing a set of labels with possible dependencies between them. Let Λ be the set of all possible labels (the domain of the y_i), and $\mathcal{Y}$, the domain of the macro-label $\mathbf{y}$. Because of the interdependencies between the y_i, some combinations of labels will not be possible, and $\mathcal{Y}$ is only a subset of Λ^n. Intuitively, the smaller $\frac{\#\{\mathcal{Y}\}}{\#\{\Lambda^n\}}$[1] is, the more regularity in the output there is and the more dependencies can help to predict the label of a variable.

[1] We use $\#\{\mathcal{A}\}$ to denote the cardinal of the set $\mathcal{A}$.

2.2 Existing Methods for Sequence Labeling: Local Output Dependencies

Many machine learning models have been proposed to take advantage of the information conveyed by label dependencies. In practice, most of them rely on local hypothesis on the label dependencies. For the sequence labeling task, a popular model is Conditional Random Fields [2]. CRFs will be used for comparison in the evaluation in section 4. More recently, the prediction of structured outputs has motivated a series of new models [3,1,4]. The SVM'ISO family of models [3], for example, is a generalization of Support Vector Machines designed for predicting structured outputs. Both CRFs and SVM'ISO consider sequence labeling as a generalization of multi-class classification: they aim at determining the macro-label $\mathbf{y}$ which is the most *compatible* with a given specific sequence of observations $\mathbf{x}$. The compatibility between the observation and the macro-label is evaluated by a θ-parametrized scoring function $F(\mathbf{x}, \mathbf{y}; \theta)$. The task of sequence prediction then amounts at finding the most compatible output among all legal outputs $\mathcal{Y}$:

$$\mathbf{y}^* = \underset{y \in \mathcal{Y}}{\operatorname{argmax}} F(\mathbf{x}, \mathbf{y}; \theta) \tag{1}$$

The argmax operator denotes the search in the space of all possible outputs $\mathcal{Y}$ that takes place during inference. Several methods have been developed to estimate the parameters θ that either optimize the conditional likelihood (in the case of CRFs) or optimize a maximum margin criterion (in the case of SVM'ISO).

In their general formulation, both CRFs and SVM'ISO can describe arbitrary dependencies. But, in practice, due to the complexity of inference and parameter estimation, the scoring function F has to be *decomposable*: a function is said to be decomposable if it can be expressed as a product of local scoring functions. The decomposition used in CRFs for sequence labeling is the following:

$$F(\mathbf{x}, \mathbf{y}; \theta) = \prod_{i=1}^{n} f(y_{i-1}, y_i, \mathbf{x}; \theta) \tag{2}$$

where f is the local scoring function, which, in the case of CRFs, is chosen to be $f(y_{i-1}, y_i, \mathbf{x}; \theta) = \exp \langle \theta, \phi(y_{i-1}, y_i, \mathbf{x}) \rangle$, where $\langle \cdot, \cdot \rangle$ is the standard dot product and ϕ is the feature vector[2]. In this decomposition, only the interactions between contiguous labels are taken into account. This allows the use of the efficient Viterbi algorithm for inference and limits the number of parameters to be estimated. The SVM'ISO family of algorithms, has been developed for modeling general output dependencies. When used for sequence labeling, however the scoring function is closely related to the one used for CRFs.

While this factorized form is critical in enabling models to work on real data, it precludes any possibility of taking non-local dependencies into account for two

[2] Compared to the usual presentation of CRFs [2], we have: $p(\mathbf{y}|\mathbf{x}; \theta) = \frac{1}{Z(\mathbf{x})} \cdot F(\mathbf{y}, \mathbf{x}; \theta)$. $Z(\mathbf{x})$ is a normalizing function that allows us to give a probabilistic interpretation to the CRF.

reasons. Firstly, in the Viterbi algorithm, the output is built by incrementally extending a partial solution towards a complete solution. As a label is often chosen before the other labels involved in the dependency are known (i.e. before the dependency can be "evaluated"), non-local dependencies cannot be exploited. Secondly, in this decomposition, a dependency is modeled by a local scoring function that has as many parameters as elements involved in the dependency. For instance, to describe a second-order dependency, we need a local scoring function with two parameters like $f(y_i, y_{i-1})$. The local scoring function has to be defined for each possible labeling of these elements. Consequently, to describe a dependency between n elements that can take m labels, m^n parameters have to be estimated, possibly resulting in sparse data problems.

Exploiting non-local dependencies therefore requires both an alternative inference algorithm and an alternative modeling of the dependencies.

2.3 Existing Methods for Sequence Labeling: Long-Term Output Dependencies

Different approaches have been proposed for taking advantage of long-range dependencies.

The *N-Best* method combined with reranking of the selected solutions [4] is a general strategy which offers an approximate solution to the structured prediction problem. It allows the use of non-local dependencies by separating the structure prediction in two independent steps. In a first step considering only local features, a limited set of potential candidates are generated with a dynamic programming algorithm. In a second step (reranking), considering arbitrary features, the "best" solution is chosen among all these candidates. Note that a limitation of reranking strategies is that the correct answer is not guaranteed to be contained in the set of potential candidates.

Several works have proposed alternatives to the Viterbi inference algorithm. Popular choices among alternative inference procedures include Gibbs sampling [9] or loopy belief propagation [5]. Another alternative is [10] which replaces the Viterbi algorithm by Integer Linear Programming to tackle the cases of non-local dependencies. In [9] and in [5], long-range dependencies are used to include domain-specific prior-knowledge, namely the so-called *label consistency* which ensures that identical observations in the corpus get the same label (e.g. in as information extraction task, it ensures that Paris is always recognized as a town). In [10] long-range dependencies are described by hand-crafted Boolean functions like "at least one element is assigned a label other than O". In all these works, the non-local dependencies are hand-crafted.

Another popular idea consists in capturing interaction among labels in a hierarchical approach [7,8]. For instance, in an Information Extraction task, [8] proposes to use a Context Free Grammar to escape the "linear tyranny" of chain-models. Within this framework, inference amounts to syntactic parsing. There are two main drawbacks in this approach: firstly, the grammar describing the interactions among labels is constructed by an expert; secondly, inference done by the CKY algorithm (a generalization of the Viterbi algorithm to trees)

has a complexity in $\mathcal{O}(n^3)$, where n is the number of elements in the sequence. A CFG has been used for comparison as a baseline non local method in the experiments described in section 4.

3 Proposed Approach

Sequence labeling with interdependent labels amounts to identifying the best assignment to a collection of variables, so that this assignment is *consistent* with a set of dependencies or a *structure*. The dependencies between outputs can be thought of as constraining the output space. We build on this idea and propose to model the dependencies with constraints (logical relations among several elements and their value), rather than by local scoring function as in the approaches discussed in Section 2.2. Typical examples of such constraints are "the label of the i^{th} variable has to be λ, if the $(i-2)^{\text{th}}$ variable is labeled by μ" or "there should be at least one variable labeled with ξ". We will associate to each constraint a weight to be interpreted as a confidence or a level of preference. More precisely, we treat sequence labeling as a *constrained assignment* problem. Let $\mathcal{V} = \{v_1, ..., v_n\}$ denote the variables describing the labels of an input sequence $\{x_1, ..., x_n\}$; each v_i may take its value in the set of labels Λ. We aim at assigning a label to each variable while satisfying a set of constraints, automatically learned from the training set.

To solve this constrained assignment problem, we propose a two-step process as advocated by [11] that relies on a well-known constraint satisfaction algorithm, relaxation labeling [13,14]: firstly, a *local classifier* affects an initial assignment to elements regardless of their context (i.e. without considering any dependencies) and, secondly, the relaxation process applies successively the constraints to propagate information and ensure global consistency.

In the following sections we detail these two steps and describe how constraints can be automatically learned from the training set. Eventually, we explain how this approach offers a solution to the problems discussed in Section 2.

3.1 Local Classifier

The local classifier aims at estimating, for each variable, a probability distribution over the set of labels. To produce these estimates, we adopt here the maximum entropy framework [12]. Note that any classifiers that output a probability distribution over the set of labels could be used as well.

Maximum entropy classifiers model the joint distribution of labels and input features. The probability of labeling a variable v with label λ is modeled by an exponential distribution:

$$p(\lambda|v;\theta) = \frac{1}{Z_\theta(v)} \exp \langle \theta, \phi(v,\lambda) \rangle$$

where $\phi(v,\lambda)$ is the feature vector describing jointly variable v and label λ, $Z_\theta(v)$ is a normalizing factor and θ the parameter vector. To estimate θ, the maximum

entropy framework advocates to choose, among all the probability distributions that satisfy the constraints imposed by the training set, the one with the highest entropy, that is to say the one that is maximally noncommittal with regard to missing information [12].

3.2 Relaxation Labeling

Relaxation Labeling (RL) [13,14] is an iterative optimization technique that solves efficiently the problem of assigning labels to a set of variables that satisfy a set of constraints. It aims at reaching an assignment with maximal consensus among the set of labels, that is to say, to assign a label to each variable while satisfying as many constraints as possible. We denote $\mathcal{V} = \{v_1, ..., v_n\}$ the set of n variables, Λ will be the set of m possible labels, and λ and μ two elements of Λ.

In the following, we assume that interactions between labels are described by a *compatibility matrix* $R = \{r_{ij}(\lambda, \mu)\}$[3]. Each coefficient $r_{ij}(\lambda, \mu)$ represents a *constraint* and measures to which extent we want to label the i^{th} variable with λ when knowing that the label of the j^{th} variable is μ: the higher $r_{ij}(\lambda, \mu)$, the more we want to label v_i with λ when the label of v_j is μ. These coefficients are estimated from the training set as detailed in Section 3.3.

The iterative algorithm of relaxation labeling works as follows: starting from an initial label assignment computed from the local classifier, the relaxation process iteratively modifies this assignment so that the labeling globally satisfies the constraints described by the compatibility matrix as well as possible. All labels are updated in parallel using the information provided by the compatibility matrix and the current label assignment.

More precisely, the local classifier defines, for each variable $v_i \in \mathcal{V}$ an initial probability vector, $\bar{p}_i^0$, with one component for each label of Λ. Let $\bar{p}_i^{(t)}(\lambda)$ denote the component of $\bar{p}_i^{(t)}$ corresponding to the label λ. Each $p_i^{(t)}(\lambda)$ describes the current confidence in the hypothesis "the label of the i^{th} variable is λ". The set $\bar{p} = \{\bar{p}_1, ..., \bar{p}_n\}$ is called a *weighted label assignment.*

Let us define, for each variable v_i and each label λ a *support function.* This function describes the compatibility of the hypothesis "the label of v_i is λ" and the current label assignment of other variables. It is generally defined by:

$$q_i^{(t)}(\lambda; \bar{p}) = \sum_{j=1}^{n} \sum_{\mu \in \Lambda} r_{ij}(\lambda, \mu) p_j^{(t)}(\mu) \tag{3}$$

Intuitively, the more confident we are in the labelings that support the hypothesis "the label of v_i is λ", the higher the support of this hypothesis (i.e. the higher $q_i(\lambda; \bar{p})$). Hypothesis we are not confident in (i.e. the ones for which $p_i(\lambda)$ is small) have only little influence. A natural way to update the weighted assignment is

[3] In this presentation, for simplification, we only consider pairwise dependencies. The extension to dependencies between an arbitrary number of dependencies is straightforward.

therefore to increase $p_i(\lambda)$ when $q_i(\lambda)$ is big, and decrease it otherwise. More precisely, the update of each $p_i(\lambda)$ is defined by:

$$p_i^{(t+1)}(\lambda) \leftarrow \frac{p_i^{(t)}(\lambda) \cdot q_i^{(t)}(\lambda, \bar{p}^{(t)})}{\sum_{\mu \in \Lambda} p_i^{(t)}(\mu) \cdot q_i^{(t)}(\mu, \bar{p}^{(t)})} \tag{4}$$

The denominator is just a normalizing factor that ensures $p_i^{(t+1)}(\lambda)$ remains a probability. This process (the calculation of the support and the update of the mapping) is iterated until convergence (i.e.:until $\bar{p}^{(t+1)} = \bar{p}^{(t)}$).

It can be proved [13,14] that, under mild assumptions, the relaxation algorithm finds a local maximum of the *average local consistency function* defined as the average support received by each variable. The latter measures the compatibility between each hypothesis "the label of v_i is λ" and all the other assignments: relaxation labeling can be seen as a method that employs the labelings we are the most confident in to disambiguate those with low confidence. The complexity of the relaxation labeling process is linear with respect to the number of variables to label.

3.3 Learning the Constraints

In some applications, the constraints are provided by hand [10] or can be easily derived from the problem specification. Here, they will be learned from the training set. Let us first observe that relevant constraints should reduce the labeling ambiguity and that choosing these constraints can however quickly become computationally intractable as the number of possible dependencies in a sequence grows exponentially with the length of the sequence.

To efficiently select the most relevant constraints, we will take advantage of the following observation: the compatibility coefficients used in the relaxation labeling process can be interpreted as *association rules*. For instance, the compatibility coefficient $r_{ij}(\lambda, \mu)$ can also be interpreted as the rule $v_j = \mu \Rightarrow v_i = \lambda$: both of them mean that, if the label μ appears at the j^{th} position of a sequence, we have a good chance of finding the label λ at the i^{th} position. Higher order dependencies are described by conjunction of label assignments. This connection between the compatibility coefficient used in relaxation labeling and association rules is appealing, since, intuitively, knowing which labels frequently co-occur in the training set, helps reducing the uncertainty of the labeling decisions.

We will draw on this intuition and consider that the compatibility coefficients are to be defined as the conditional entropy [15] of the corresponding association rule. The conditional entropy of an association rule is a combination of the usual *support* and *confidence* used to evaluate the importance of a rule. Combinatorial algorithms, such as Apriori [16] can be used to extract efficiently all the association rules the conditional entropy of which is higher than a user-provided value. This value is a parameter of our algorithm. Note that while Apriori is a combinatorial method, it is used here during training and inference remains linear wrt the sequence size.

For simplicity, relaxation labeling was described using absolute i, j positions. The algorithm is actually implemented using relative variable positions which are more general and flexible. An example of a rule expressed with relative value is $v_{i-3} = \alpha \Rightarrow v_i = \beta$. Relaxation labeling can be easily generalized to handle arbitrary rules. The only difference relies on the definition of the support (Equation 3) that has to be generalized to consider all elements that appear in the rule. For instance, if we consider the previous rule and a sequence of 9 elements in which $v_1 = \alpha$ and $v_5 = \alpha$, the hypothesis $v_4 = \beta$ and $v_8 = \beta$ will be both strengthened. In the same way, association rules are learned with relative values and are then instantiated in any position that fits elements in a sequence.

3.4 Advantages of Our Approach

Our approach solves several of the restrictions that were pointed out in Section 2. First, the most likely labeling is inferred by iteratively reassigning labels based on the current assignment of all other variables, contrary to chain-models in which only previous assignments are taken into account. As a result, each label is chosen according to a global context and dependencies describing the sequence as a whole or involving non-sequential variables can be considered. Relaxation labeling is therefore an interesting alternative to the Viterbi algorithm for sequence labeling.

Secondly, as they are formed by variable-value pairs (v_i, λ), constraints can account for dynamically changing dependencies, which are conditioned on the values of variables. Consequently, sparse data problems are avoided: in our framework only the variable values involved in a constraint have to be considered, while, for the models described in Section 2, the score of all possible assignments to these variables has to be estimated This representation also allows us to efficiently select the relevant dependencies we want to incorporate in the model.

4 Empirical Evaluation

4.1 Tasks and Corpora

We have tested our approach on two different sequence labeling tasks: structured data extraction [17] and chunking [18].

Structured Data Extraction. The first application considered, structured data extraction [19,20,6], is motivated by the development of the *Semantic Web*. Semantic Web aims at providing value-added services by taking advantage of a semantically-rich structured view of HTML or XML documents instead of their traditional bag-of-words representation. The semantic technologies need hard-wiring knowledge of the structure they are using and, therefore, can only deal with documents that comply strictly with a *schema*. This schema, generally

expressed by a DTD or an XML Schema, defines, a priori, allowed structures of documents. Because of the lack of standardization, the representation of data varies from source to source, and, consequently, the deployment of the Semantic Web is only possible if we are able to resolve these heterogeneities. One first step towards this resolution consists in *extracting* relevant information from structured documents.

Structured data extraction is closely related to the standard task of information extraction, but it is made easier by the presence of a document structure [6]. We consider this task here as a sequence labeling task: structured data extraction amounts to labeling a sequence of observations defined by the leaves of an HMTL or XML document, where dependencies between the labels are described by a *target schema*.

Our model has been tested on two different corpora. The first one is the collection `Courses` used in [20] that describes lectures in five different universities. There are more than 12,000 descriptions which are annotated with 17 different labels such as `lecturer`, `title`, `start time` or `end time`; each description contains between 4 and 552 elements to be extracted. The second corpus, `MovieDB` is based on the IMDb database [6]. There are 4,483 movie descriptions, annotated with 16 different labels such as `actor`, `director` or `title`.

Considering non-local dependencies between the labels is mandatory to achieve good performances in this task. Indeed, in many cases the local classifier does not have enough information to choose the correct label and dependencies have to be considered to reduce labeling ambiguities. For instance, the only way to distinguish a `start time` from an `end time` or `actor` from a `director` is by taking into account the context, i.e. the position of the element in the sequence of labels.

Each collection was randomly split in two equal parts for training and testing. Experiments were performed on the two corpora using the same features for the local classifier described in Section 3.1. We used the kind of features generally used in information extraction tasks: typical examples of these features include `NumberOfWords`, `NumberOfCapitalLetter` or `ContainsHTTP`.

Chunking. The second application we considered is the "All-phrase chunking" task of CoNLL 2000 [18]. Text chunking aims at identifying the non-recursive cores of various phrase type in text. There are 11 different chunks in the corpora such as "noun phrase", "adjective phrase" or "subordinated clause". The chunks are represented with three kinds of labels: "B-X" stands for "first word of a chunk of type X", "I-X" for "non-initial word in an X chunk" and "O" means "word outside of any chunk". Using this so-called BIO representation allows to tackle a sequence segmentation task[4] as a sequence labeling task. As pointed out by [10] and [18], the sequences of labels involved in this task present many dependencies, and the BIO representation naturally forbids some combinations of labels. In our experiments we used the features and the train set and test set provided by [18].

[4] when several consecutive elements of the observation sequence are mapped to a single label.

4.2 Results

Baseline Models. As baseline models we used a Maximum Entropy classifier (the local classifier described in Section 3.1), standard linear-chain CRF[5] and a grammar-based extraction approach similar to the ones presented by [8] or [22]. Because of its computational complexity, the SVM'ISO approach we described in Section 2.2 cannot be used on our corpora. The local classifier does not incorporate any information about the dependencies between labels, the CRF only considers local and sequential dependencies (see Section 2) while the grammar-based approach can take long-range dependencies into account.

The principle of this latter approach is as follows: a Maximum Entropy Classifier estimates a probability distribution over the set of labels for each observation and a Probabilistic Context Free Grammar is then used to infer a tree structure. This tree structure can be seen as a hierarchy of hidden variables that describes long-range dependencies. The predicted label sequence is defined by the labels of the leaves of the tree with the highest score. The inference complexity of this approach is $\mathcal{O}(n^3)$, where n is the number of elements to be labeled, which should be contrasted with the complexity $\mathcal{O}(n)$ of both CRFs and our method. This model requires us to define a context-free grammar that describes both local and non-local dependencies. This can be done in the structured extraction task by converting the target schema in context-free grammar [22,6], but not in the chunking task, where no general grammar that describes interactions between chunks is known.

Results. Table 1 presents the results of the different experiments. The scores presented correspond to the standard F1 measure.

Results show the importance of taking into account the dependencies: in all the tasks, the score of the local classifier is always the worst. As was explained in Section 4.1, this is mainly due to the fact that in many cases, an observation does not contain enough information to choose the correct label so that the context has to be considered. Exploiting non-local dependencies is also of great help. On the data extraction task both our approach and the grammar-based approach clearly outperform CRFs. In the chunking task, CRFs achieve slightly better performances, likely because the dependencies between labels are less relevant, but both learning and inference are much faster with our method than with CRFs. The grammar-based approach and our approach achieve similar performances, which shows the ability of the proposed method to select relevant dependencies. Note that inference with our method is also an order of magnitude faster than with a grammar-based approach.

In the experiments, CRFs were used with default parameters and better results might be obtained by tuning these parameters for the different tasks. CRFs results on the Course corpus are particularly low. This is due to regularization in learning: to avoid estimation problem, parameters are smoothed so that the weight of each possible transition between two successive nodes is non-zero. All transitions are thus allowed which does not reflect correctly the structure of the data (For course data, most transitions should be 0).

[5] In our experiments we used FlexCRF [21].

Table 1. Results of the different experiments. Dashes indicates that the experiments could not be performed. The reported score corresponds to the standard F1 measure.

	MovieDB	Course	Chunking
Local Classifier	90.6%	47.9%	90.3%
Proposed Model	**97.4%**	**88.1%**	**93.2%**
CRF	96.4%	78.7%	**94.6%**
Grammar	**97.5%**	**87.4%**	—

5 Conclusion

We have proposed a general method for efficiently discovering relevant non-local dependencies and exploiting them in the sequence labeling task. Our approach relies on the modeling of relationships between labels by constraints. It is a two steps process: initial label assignment is provided by a local classifier, the dependencies among variables are considered and propagated using an iterative relaxation procedure.

This model can account for dynamically changing dependencies. This is in contrast to most existing approaches that assume that a fixed-sized neighborhood is relevant for predicting each label. The proposed approach has achieved convincing results on different tasks at a low computational cost. A key element in developing such approaches is to define measures to assess the strength of a dependency and the amount of information a dependency provides to reduce labeling ambiguity.

Acknowledgments

We thank Nicolas Usunier and Alexander Spengler for enlightening discussions and insightful comments on early drafts of this paper. This work was supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778. This publication only reflects the authors' views.

References

1. Taskar, B., Lacoste-Julien, S., Jordan, M.I.: Structured prediction, dual extragradient and bregman projections. Journal of Machine Learning Research (2006)
2. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: ICML'01, pp. 282–289. Morgan Kaufmann, San Francisco, CA (2001)
3. Tsochantaridis, I., Joachims, T., Hofmann, T., Altun, Y.: Large margin methods for structured and interdependent output variables. Journal of Machine Learning Research 6, 1453–1484 (2005)
4. Collins, M., Koo, T.: Discriminative reranking for natural language parsing. Computational Linguistics 31, 25–69 (2005)

5. Sutton, C., McCallum, A.: Collective segmentation and labeling of distant entities in information extraction. In: ICML workshop on Statistical Relational Learning (2004)
6. Wisniewski, G., Gallinari, P.: From layout to semantic: a reranking model for mapping web documents to mediated xml representations. In: 8th RIAO International Conference on Large-Scale Semantic Access to Content (2007)
7. Awasthi, P., Gagrani, A., Ravindran, B.: Image modeling using tree structured conditional random fields. In: IJCAI (2007)
8. Viola, P., Narasimhan, M.: Learning to extract information from semi-structured text using a discriminative context free grammar. In: SIGIR '05 (2005)
9. Finkel, J., Grenager, T., Manning, C.D.: Incorporating non-local information into information extraction systems by gibbs sampling. In: ACL'05 (2005)
10. Roth, D.: Integer linear programming inference for conditional random fields. In: ICML, pp. 736–743. ACM Press, New York (2005)
11. Punyakanok, V., Roth, D., tau Yih, W., Zimak, D.: Learning and inference over constrained output. In: IJCAI'05, pp. 1124–1129 (2005)
12. Berger, A.L., Pietra, V.J.D., Pietra, S.A.D.: A maximum entropy approach to natural language processing. Comput. Linguist. 22(1), 39–71 (1996)
13. Pelillo, M.: The dynamics of nonlinear relaxation labeling processes. J. Math. Imaging Vis. 7(4), 309–323 (1997)
14. Hummel, R.A., Zucker, S.W.: On the foundations of relaxation labeling processes. IEEE PAMI 5(1), 267–287 (1983)
15. Blanchard, J., Guillet, F., Gras, R., Briand, H.: Using information-theoretic measures to assess association rule interestingness. In: ICDM'05., pp. 66–73 (2005)
16. Borgelt, C.: Efficient implementations of apriori and eclat. In: Workshop of Frequent Item Set Mining Implementations (FIMI) (2003)
17. Liu, B., Chen-Chuan-Chang, K.: Editorial: special issue on web content mining. SIGKDD Explor. Newsl. 6(2), 1–4 (2004)
18. Sang, E.F.T.K., Buchholz, S.: Introduction to the conll- 2000 shared task: chunking. In: 2nd workshop on Learning language in logic and the 4th conference on Computational natural language learning, Morristown, NJ, USA pp. 127–132 (2000)
19. Shadbolt, N., Berners-Lee, T., Hall, W.: The semantic web revisited. IEEE Intelligent Systems 21(3), 96–101 (2006)
20. Doan, A., Domingos, P., Halevy, A.: Learning to match the schemas of data sources: A multistrategy approach. Mach. Learn. 50(3), 279–301 (2003)
21. Phan, X.H., Nguyen, L.M.: Flexcrfs: Flexible conditional random field toolkit (2005), `http://www.jaist.ac.jp/~hieuxuan/flexcrfs/flexcrfs.html`
22. Chidlovskii, B., Fuselier, J.: A Probabilistic Learning Method for XML Annotation of Documents. In: IJCAI (2005)