BiLSTM-SSVM: Training the BiLSTM with a Structured Hinge Loss for Named-Entity Recognition

Hanieh Poostchi, *Student Member, IEEE,* Massimo Piccardi, *Senior Member, IEEE*

Abstract—Building on the achievements of the BiLSTM-CRF in named-entity recognition (NER), this paper introduces the BiLSTM-SSVM, an equivalent neural model where training is performed using a structured hinge loss. The typical loss functions used for evaluating NER are entity-level variants of the F_1 score such as the CoNLL and MUC losses. Unfortunately, the common loss function used for training NER - the cross entropy - is only loosely related to the evaluation losses. For this reason, in this paper we propose a training approach for the BiLSTM-CRF that leverages a hinge loss bounding the CoNLL loss from above. In addition, we present a mixed hinge loss that bounds either the CoNLL loss or the Hamming loss based on the density of entity tokens in each sentence. The experimental results over four benchmark languages (English, German, Spanish and Dutch) show that training with the mixed hinge loss has led to small but consistent improvements over the cross entropy across all languages and four different evaluation measures.

Index Terms—Natural language processing, named-entity recognition, sequential labeling, loss-augmented inference, non-decomposable evaluation loss.

1 Introduction

The main aim of classifier training is to find a parametrization minimizing the expectation of a chosen loss function. However, the loss functions commonly used for evaluation such as the 0-1 loss are discontinuous in parameter space, non-convex and flat over large regions. For this reason, the common approach is to optimize an alternative function, referred to as *surrogate loss*, instead of the chosen loss. The most well-known surrogates are the logistic loss and the hinge loss which are both upper bounds of the 0-1 loss. In the deep learning community, the logistic loss is also known as *cross entropy* (or negative log-likelihood) and is the de facto objective for training.

The typical functions used for evaluating named-entity recognition (NER) and other sequential labeling tasks are derivatives of the F_1 score and include the CoNLL and MUC scores [1]. The relationship of the corresponding losses (i.e., the negated scores) with the cross entropy is loose. However, the hinge loss can still be defined as a formal upper bound for all of them. For this reason, in this paper we propose an approach for training NER using a hinge loss that bounds the CoNLL loss from above.

Given its inherently sequential nature, NER is often tackled by sequential classifiers such as linear-chain conditional random fields and recurrent neural networks. In particular, the BiLSTM-CRF [2], [3] is a high-performing deep learning architecture obtained from the combination of a long short-term memory (LSTM) and a conditional random field (CRF) as the output layer. The addition of the CRF permits efficient, structured prediction of the entire sequence of labels and tends to increase the classification accuracy [2], [3], [4]. Following the achievements of the BiLSTM-CRF, in this paper we propose the *BiLSTM-SSVM*, an equivalent neural model where training is performed using the structural support vector machine (SSVM) [5]. The crux of this method is the solution of a special inference problem known as the "loss-augmented inference". This inference returns the sequence of labels maximizing the sum of the score and the chosen loss, and it is needed in order to ensure that the training objective acts as an upper bound on the loss. The loss-augmented inference is straightforward if both the scoring and loss functions decompose over the individual labels in the sequence: however, this is not the case for the CoNLL loss and any other entity-based loss. Therefore, this paper presents a novel dynamic programming algorithm to address this case. Overall, our main contributions are:

- A training approach for the BiLSTM-CRF based on the minimization of a structured hinge loss (BiLSTM-SSVM). This loss can usefully bound the evaluation losses commonly employed for NER such as CoNLL and MUC;
- A dynamic programming algorithm for the loss-augmented inference with the CoNLL loss. This algorithm is presented in Section 4.1 and a proof of optimality is given in Section 4.3;
- Experimental results on NER over four benchmark languages (English, German, Dutch and Spanish) showing that the proposed approach has led to slight yet consistent improvements over the conventional cross-entropy training, across all the tested languages and four different evaluation measures (CoNLL, MUC, en-

• *Both authors are with the School of Electrical and Data Engineering, Faculty of Engineering and Information Technology, University of Technology Sydney (UTS), Ultimo, NSW, 2007, Australia.*
E-mail: hanieh.poostchimohammadabadi@uts.edu.au, massimo.piccardi@uts.edu.au

Manuscript received February 06, 2019; revised June 15, 2019.

tity segmentation F_1 score and entity classification F_1 score).

The rest of this paper is organized as follows: Section 2 describes the main related work. Section 3 recaps sequential labeling and the BiLSTM-CRF. Section 4 introduces the BiLSTM-SSVM and presents the proposed algorithm for the loss-augmented inference under the CoNLL loss. Further, it introduces a mixed hinge loss combining the CoNLL and Hamming losses as training objectives. Section 5 describes the experiments and results. Eventually, Section 6 concludes the paper.

2 Related Work

Most of the NER approaches proposed in recent years leverage recurrent neural networks (RNNs), and a selection is briefly reviewed in the following. One of the main initial works in this area is [6] that proposed an architecture made of a convolutional network and a CRF output layer to be used for chunking, POS tagging and NER. The model is trained by using stochastic gradient ascent to maximize the cross entropy, and an implicit word embedding is automatically learned in the network's early layers from a random or external initialization. Recurrent neural networks such as the Elman and Jordan RNNs have also been used for sequential labeling [7], [8]. The Elman RNN is similar to the feed-forward neural networks, except that the output of the hidden layer at slot $t-1$ is fed back into the input at slot t. Conversely, in the Jordan RNN it is the output layer that feeds back into the input.

To capture the properties of both RNNs and CRFs for sequential tagging, [2] have combined the bidirectional LSTM with a CRF output layer. In this model, the LSTM is used first to process each sentence token-by-token and produce an intermediate representation. Then, the intermediate representation is used as input for the CRF to provide the joint prediction of all the labels. [3] have extended this model with a second, auxiliary LSTM encoding each token character-by-character to capture the regularities at character level. This extended model, that we refer to simply as BiLSTM-CRF in the following, had reported state-of-the-art NER accuracy for several benchmark languages at the time. Since then, it has been adopted widely as a strong baseline for comparison. Variants have also been proposed such as the BiLSTM-CNN-CRF [4] which is a hybrid network where the auxiliary LSTM is replaced by a CNN.

More recently, sequential labeling has extensively incorporated neural language models (LM) to improve the token encoding. [9] have proposed TagLM, a hierarchical RNN model where pre-trained, bidirectional LM embeddings are concatenated with the output of the first bidirectional RNN layer. In ELMo [10], the bidirectional LM embeddings are obtained from the aggregation of all the internal layers of a deep bidirectional LM. [11] have proposed the LM-LSTM-CRF, where transfer learning and multi-task learning are used to extract character-level representations. In this model, the objective functions of the LM and the sequence labeling are minimized jointly. In a similar vein, in [12] the objective function includes the prediction of the previous word, the current label and the next word in the sequence.

As an alternative to the cross entropy, the hinge loss has been used as the training objective in a number of works. The Recurrent SVM is an LSTM trained using an SVM objective [13], where the parameters of the LSTM and the SVM are learned jointly using a combination of sequence-level and frame-level regularized hinge losses. The model has been evaluated on the Windows phone task for speech recognition and has yielded improvement over a standard LSTM. The RSVM, a combination of RNN and structured SVM, has been proposed in [14] for slot tagging in spoken language understanding. This network improves the discriminative capability of an RNN by optimizing a sequence-level max-margin training criterion [15]. The training times take advantage of the fact that the hinge loss can be an identical zero for some training samples, and therefore such samples do not need to be involved in the parameter updates. Similarly, [16] has proposed the DeepSegmentor, a neural model for speech segmentation composed of an RNN and a structured prediction output layer that are trained jointly using a structured loss function (the combined duration).

The loss-augmented inference, too, has received significant attention in the literature. In an early work, [5] proposed SVM^{perf}, a support vector method for loss functions such as the Hamming loss, the F_1 loss, the precision-recall break-even point, the precision and recall at k, and the ROC area. However, this method only addresses independent and identically distributed (i.i.d.) data, and the extension to structured prediction is challenging because of the dependencies within the scoring function. For the structured prediction case, [17]) have proposed a training algorithm for the F_1 loss that approximates the evaluation loss with a softmax; [18] have proposed an approximate inference algorithm for the AUC loss based on a linear programming relaxation; [19] have proposed an approximate algorithm for the F_1 loss leveraging a dual decomposition and alternate optimizations; and [20] have proposed an exact, exhaustive algorithm for the loss-augmented inference under the F_1 loss, yet not for the multi-class case required by NER. Closely inspired by this algorithm, in this paper we instead propose an algorithm for multi-class loss-augmented inference under the CoNLL loss.

Other structured surrogate loss functions that have found use in the literature include the structured probit loss [21], the structured ramp loss [21], [22], [23] and the structured orbit loss [24]. Other work has proposed the direct minimization of the evaluation loss by means of asymptotic equalities [25], [26]. However, their adoption to date has been more limited.

3 Sequential Labeling

Sequential labeling aims to predict a sequence of class labels, $y = \{y_1 \dots y_t \dots y_T\}$, from a corresponding sequence of measurements, $x = \{x_1 \dots x_t \dots x_T\}$. It is a very common task in NLP for applications such as chunking, POS tagging, supertagging and NER, where the sequence of measurements is typically a sentence. A widespread model for sequential labeling is the hidden Markov model (HMM) that factorizes the joint probability of the measurements and the labels, $p(x, y)$, by arranging the latter in a Markov chain

(of order one or above) and conditioning the measurement at frame t on only the corresponding label. For an HMM of order one, $p(x, y)$ is expressed as:

$$p(x, y) = p(y_1) \prod_{t=2}^{T} p(y_t|y_{t-1}) \prod_{t=1}^{T} p(x_t|y_t) \tag{1}$$

where $p(y_1)$ is the probability of the initial class, terms $p(y_t|y_{t-1})$ are the transition probabilities and terms $p(x_t|y_t)$ are the emission, or measurement, probabilities. With limited modifications (exponential family for the emission probabilities, denormalized factors), this model becomes the widely-used CRF, allowing for discriminative training [27].

3.1 The BiLSTM-CRF

The BiLSTM-CRF is a recurrent neural network obtained from the combination of an LSTM and a CRF [2], [3]. These two models enjoy complementary features: as a complex, nonlinear model, the LSTM can effectively capture the sequential relationships amongst the input tokens. In turn, the CRF permits optimal, joint prediction of all the labels in the sentence, capturing the relationships at label level. In the BiLSTM-CRF, the posterior probability of label sequence y given input sequence x can be expressed as:

$$p(y|x) = \frac{\exp(F(x, y))}{\sum_{u \in Y} \exp(F(x, u))} \tag{2}$$

where $F(x, y)$ is the scoring function of the BiLSTM-CRF and Y is the set of all possible predictions. Given a training set of labelled sequences, $\{x_i, y_i\}, i = 1 \dots N$, the BiLSTM-CRF can be trained by minimizing the cross entropy (or negative conditional log-likelihood):

$$\bar{w} = \underset{w}{\operatorname{argmin}} - \sum_{i=1}^{N} \ln p(y_i|x_i, w) \tag{3}$$

where we have made explicit the dependence on the model's parameters, w, including the transition weights of the CRF, the weights of the main and auxiliary LSTMs, and the embeddings of all tokens and characters. Replacing (2) in (3), we obtain:

$$\begin{aligned} \bar{w} = & \underset{w}{\operatorname{argmin}} \\ & - \sum_{i=1}^{N} [F(x_i, y_i; w) - \ln \sum_{u \in Y} \exp(F(x_i, u; w))] \end{aligned} \tag{4}$$

showing that the optimal parameters are a trade-off between the score assigned to the true labeling and the log-sum-exp of the scores of all the possible labelings. Once the model is trained, inference for a new sentence x is obtained as:

$$\bar{y} = \underset{y}{\operatorname{argmax}}\, p(y|x, \bar{w}) = \underset{y}{\operatorname{argmax}}\, F(x, y; \bar{w}) \tag{5}$$

by propagating x through the network and applying the Viterbi algorithm at the CRF output layer.

4 The proposed BiLSTM-SSVM

Most NLP tasks rely on dedicated performance measures for evaluation. Examples include the BLEU score for machine translation, the ROUGE score for summarization, and the MUC and CoNLL scores for NER [28]. It is therefore tempting to train the classifier to explicitly minimize such evaluation losses in the hope of obtaining higher predictive accuracy [17], [19], [20]. However, a risk of bias toward certain predictions exists and the proof is ultimately empirical. Hereafter, we consider NER and adopt the surrogate loss of SSVM [5], namely the *structured hinge loss*, which can be made a convex upper bound for any chosen evaluation loss [29]. Given a training set, $\{x_i, y_i\}$, $i = 1 \dots N$, and an evaluation loss, $\Delta(y_i, y)$, the hinge loss for the i-th sample is defined as:

$$\begin{aligned} l_i &= \max_{y \in Y} \big[- F(x_i, y_i) + F(x_i, y) + \Delta(y_i, y) \big]_+ \\ &= \big[- F(x_i, y_i) + F(x_i, y_i^*) + \Delta(y_i, y_i^*) \big]_+ \end{aligned} \tag{6}$$

where $[\pi]_+ = \max\{\pi, 0\}$ and

$$y_i^* = \underset{y \in Y}{\operatorname{argmax}}\, F(x_i, y) + \Delta(y_i, y). \tag{7}$$

Equation (6) shows that the hinge loss is different from zero if the score assigned to the ground-truth labeling does not surpass that of any other labeling by an amount equal to the evaluation loss itself. The value of the hinge loss is therefore set by a "most violating" labeling that becomes the explicit target for the margin. [15] have proven that this is a sufficient condition for the hinge loss to bound the evaluation loss from above. The inference in (7) is commonly referred to as "loss-augmented inference" and is the core of structural SVM. In the case of scores and evaluation losses that can be computed token-by-token (such as the 0-1 loss or the Hamming loss), a Viterbi algorithm with appropriate weights can still be used to compute the loss-augmented inference [15]. However, there are two standing problems in using the CoNLL loss for the loss-augmented inference: 1) similarly to the F_1 loss, the CoNLL loss is based on precision and recall and is therefore non-decomposable over single tokens; and 2) the evaluation of correct and incorrect predictions inherently spans multiple tokens. Our main contribution, presented in the following, is a dynamic programming algorithm that solves the loss-augmented inference in the case of the CoNLL loss.

4.1 Loss-Augmented Inference Under the CoNLL Loss

The CoNLL score is an F_1 score specialized for the NER task. In this score, a prediction is counted as a true positive only if the entire named entity mention is segmented and classified correctly. The tokens are commonly annotated in the IOB format (which stands for *inside*, *outside*, *beginning* of an entity) and with their true class label. Since the CoNLL score is normalized between zero and one, the CoNLL loss can be naturally defined as its complement to one. In the following, we present a dynamic programming algorithm for multi-class sequential labeling that solves the loss-augmented inference of (7) under this loss.

The CoNLL loss is an evaluation loss and, as such, it only depends on the classification contingency table, i.e.,

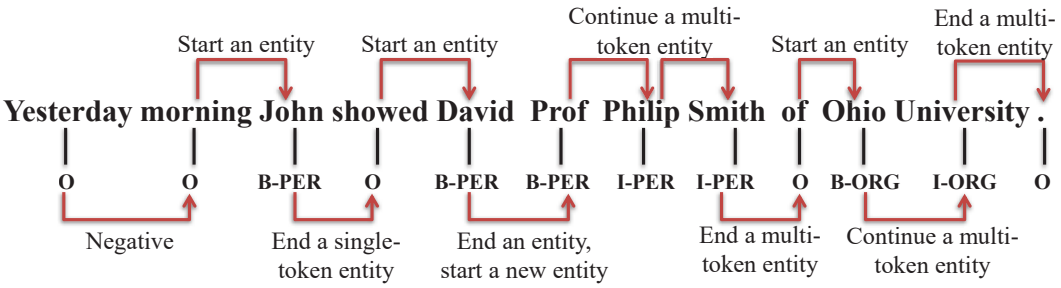

Fig. 1. A pseudo-sentence illustrating all the cases of label transitions.

the values of TP, FN, TN, and FP. Given a ground-truth labeling y^g, the number of true entities in a sentence is fixed and known, and it can be expressed as $P = TP + FN$. Accordingly, the CoNLL loss can be written as:

$$\Delta_{CoNLL}(y^g, y) = 1 - \frac{2\ pre \times rec}{pre + rec}, \quad pre = \frac{TP}{TP + FP}, \quad rec = \frac{TP}{P} \tag{8}$$

showing that (8) depends on the prediction only via the value of TP and $(TP+FP)$. The loss-augmented inference aims to find the y^* labeling with the highest sum of score and loss. Following [20], we can approach this problem in two steps: 1) finding the labelings maximizing the score alone for fixed values of TP and $(TP+FP)$, and 2) finding the best of all these labelings. Given that the loss is solely a function of TP and $(TP + FP)$, this ensures finding the required maximum.

The modified Viterbi algorithm of [20] extends the notion of Viterbi state at slot t with the TP and $(TP+FP)$ counts up to that slot. If these counts can be incremented token-by-token as in [20], the update rules of the algorithm are relatively simple. However, with the CoNLL loss a true positive prediction can only be resolved at the end of a chunk, which can span multiple tokens, and therefore the required update rules are substantially more complicated. We present the proposed dynamic programming algorithm in Algorithms 1 and 2, with a description hereafter. A proof of optimality is provided in section 4.3.

In the proposed algorithm, prediction y is developed in left-to-right order along the sequence. Thus, the TP and $(TP+FP)$ counts can only increment or remain unchanged. The state of the partial solution at slot t is indicated with $(TP, (TP + FP), y_t)$ and it consists of the predicted label for the current token, y_t, and the TP and $(TP + FP)$ counts up to the current token. The sequence itself is noted as $seq(TP, (TP + FP), y_t)$ and the scoring function for a sequence is noted as $F(seq)$. The induction step is as follows: at slot t, the partial solution is obtained by extending a number of the partial solutions at slot $t-1$ with the current prediction, y_t and the corresponding increment of TP and $(TP + FP)$. The domain for y_t is $L = \{$O, B-PER, ... B-LOC, I-PER, ... I-LOC$\}$, even in the case where this gives place to an invalid IOB annotation (for instance, an I-x label following an O one) since the run-time model predicts in a similarly unconstrained way. We use shorthand notations B-x and I-x to mean one of the B and I labels, and notations B-* and I-* to mean any of them.

All the update rules in Algorithm 1 depend on the values of the current true label, y^g_t, and the immediately previous, y^g_{t-1}. Figure 1 shows examples of possible label transitions. To facilitate an understanding of the update rules, we have explicitly named the label transitions as:

- 'Negative': transition from O to O;
- 'Start an entity': transition from O to B-*;
- 'Continue a multi-token entity': transition from B-x or I-x to I-x;
- 'End an entity, start a new entity': transition from B-* or I-* to B-*;
- 'End a multi-token entity: transition from I-* to O;
- 'End a single-token entity': transition from B-* to O.

Due to space limitations, Algorithm 1 only shows the update rules for the 'Negative' case (the complete rules are provided as Supplemental Material)[1]. In this case:

- predicting y_t as O is a true negative and increments neither TP nor $(TP + FP)$;
- predicting y_t as B-* starts a false positive and, as a consequence, it increases the $(TP + FP)$ count by one;
- predicting y_t as I-x with $y_{t-1} \in \{$B-x,I-x$\}$ means that the prediction of a multi-token entity is continuing and neither TP nor $(TP + FP)$ is incremented;
- instead, predicting y_t as I-x with $y_{t-1} \in L \setminus \{$B-x,I-x$\}$ starts a new false positive and, therefore, $(TP + FP)$ is increased by one.

Figure 2 visually summarizes the above update rules for immediacy. The update rules for all cases are fully determined by the values of y^g_t, y^g_{t-1}, y_t, and y_{t-1} and they are completely independent of the specific class set (in number and type). Therefore, the proposed algorithm can be used for any NER task. The algorithm is highly articulated since its cases stem from the combination of multiple variables: however, its computational complexity is only approximately quadratic in the length of the sentence and therefore manageable even for sentences of a few hundred tokens.

Algorithm 1 returns all the sequences of highest score for $TP = 0 \dots P, (TP + FP) = 0 \dots |y^g|$. After that, Algorithm 2 simply searches exhaustively over such best sequences for the one maximizing the sum of the score and the loss, y^*.

1. All our code and scripts are provided in the Supplemental Material.

Algorithm 1 Finding the argmax of equation (7) for every possible value of TP and $(TP + FP)$.

procedure FINDBESTSEQUENCES

Input: ground-truth sequence y^g (length: T)
Output: set of predicted sequences, one per possible value of TP and $(TP + FP)$ and end label y_T

// $seq(i, j, y_t)$: predicted sequence of length t with i true positives, j true + false positives and end label y_t
L = {O, B-PER, . . . , B-LOC, I-PER, . . . , I-LOC} // the set of possible labels
$TP_{max} = 0$, $(TP + FP)_{max} = 0$ // initializes both error limits to zero

for $t = 1 : T$ **do**
  $(TP + FP)_{max}$++ // new token: increments the possible FPs
  if END-ENTITY **then**
    TP_{max}++ // if at the end of an entity, also increments the possible TPs
  end if
  switch $(y^g_{t-1} \rightarrow y^g_t)$ **do** // choose action based on the previous and current ground-truth tokens
    case (O→O) // this case is called: **Negative**
      // this loop iterates over all the partial predicted sequences to extend them by one token:
      for i = $0 : TP_{max}$, j = $0 : (TP + FP)_{max} - 1$, **each** $y_t \in L$ **do**
        switch (y_t) **do**
          case (O)
            $seq(i, j, y_t) = \text{argmax}_{y_{t-1} \in L}\ F\Big(seq(i, j, y_{t-1}),\text{O}\Big)$ // prediction is correct: i, j unchanged
          case (B-*)
            $seq(i, j + 1, y_t) = \text{argmax}_{y_{t-1} \in L}\ F\Big(seq(i, j, y_{t-1}),\text{B-*}\Big)$ // false positive: increments j
          case (I-*)
            $seq(i, j, y_t) = \text{argmax}_{y_{t-1} \in \{\text{B-*,I-*}\}}\ F\Big(seq(i, j, y_{t-1}),\text{I-*}\Big)$ // incorrect, but j unchanged
            $seq(i, j + 1, y_t) = \text{argmax}_{y_{t-1} \in L \setminus \{\text{B-*,I-*}\}}\ F\Big(seq(i, j, y_{t-1}),\text{I-*}\Big)$ // false positive, increments j
        end for
      // all other cases follow; details in the Supplementary Material:
      case (O→B-x) ... // **Start an entity**
      case (B-x→I-x) || (I-x→I-x) ... // **Continue a multi-token entity**
      case (B-*→B-x) || (I-*→B-x) ... // **End an entity, start a new entity**
      case (I-x→O) ... // **End a multi-token entity**
      case (B-x→O) ... // **End a single-token entity**
  end for
end procedure

4.2 The Mixed Hinge Loss

Despite our emphasis, using the CoNLL loss as a training objective may not prove optimal for test-time performance. The CoNLL score is of the F_1 family and, as such, it is a meaningful performance measure when the positive samples (i.e., the named-entity mentions) are relatively few [5]. Conversely, the Hamming loss is more meaningful in more balanced cases. This suggests exploring a *mixed hinge* loss, where the CoNLL loss is minimized for sentences where the entity density is below a chosen threshold, and the Hamming loss is minimized otherwise. The entity density is simply computed as the percentage of entity tokens in the sentence, and the threshold is tuned using cross-validation.

4.3 Proof of Optimality for the Loss-Augmented Inference Algorithm

The algorithm described in Algorithms 1, 2 returns the labeling maximizing (7) under the CoNLL loss. Since a proof was not given in [20], we provide an original proof hereafter.

Proposition: Algorithms 1, 2 return the labeling maximizing loss-augmented inference (7).

Proof: Let us consider a function, $f(y)$, of labeling y, and another function, $g(s(y))$, that is a function of y only through a set of sufficient statistics, s (e.g., an evaluation loss). The sufficient statistics take value over a finite integer interval, $s \in [0 \dots N]$ for reference, and, as such, g only takes at most $N + 1$ distinct values.

Algorithm 2 Finding the argmax in equation (7) - final loop.

procedure FINALLOOP
  $best = -\infty$
  $best_{sequence} = [\,]$
  // this loop iterates over all the predicted sequences to select the one with the highest sum of score and loss:
  for $i = 0 : P$, $j = 0 : T$, **each** $y_T \in L$ **do**
    $value = F\Big(seq(i, j, y_T)\Big) + \Delta_{CoNLL}\Big(y^g, seq(i, j, y_T)\Big)$
    if $value > best$ **then**
      $best = value$
      $best_{sequence} = seq(i, j, y_T)$
    end if
  end for
end procedure

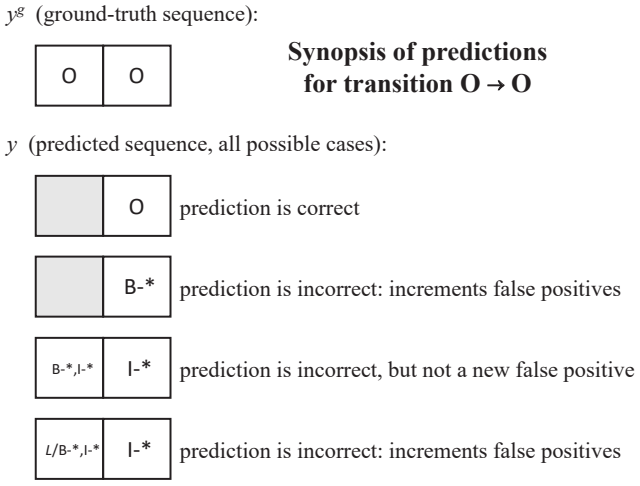

Fig. 2. A synopsis of all the possible predictions for ground-truth transition O→O ('Negative' case).

Let us now consider:

$$\begin{aligned} &\bar{y}_k = \underset{y \in Y}{\operatorname{argmax}}\, f(y) \\ &s.t.\ \ s(y) = k,\ \ \forall k \in [0 \dots N] \end{aligned} \tag{9}$$

This maximization returns a set of $N + 1$ arguments of constrained maxima, one for each value of the sufficient statistics. Therefore,

$$\bar{y} = \underset{y \in \{\bar{y}_1, \dots, \bar{y}_N\}}{\operatorname{argmax}} f(y) + g(s(y)) \tag{10}$$

returns the desired maximum. For ease of reference, in (7) $f(y)$ is scoring function $F(x_i, y)$, and $g(s(y))$ is evaluation loss $\Delta(y_i, y)$, which is a function of y only through sufficient statistics TP and $(TP + FP)$. Algorithm 1 implements (9) while Algorithm 2 implements (10). This algorithm is viable for reasonably small values of N. □

5 Experiments and Results

We have run experiments over four well-known NER datasets: English and German from CoNLL-2003 [30], and Spanish and Dutch from CoNLL-2002 [31][2].

TABLE 1
Comparison of the CoNLL scores achieved by BiLSTM-CRF as reported in [3] and those obtained by these authors with NeuroNER [32]'s BiLSTM-CRF implementation.

Experiment	English	German	Spanish	Dutch
Lample *et al.* [3]	90.94	78.76	85.75	81.74
Cross Entropy	90.412	82.156	84.312	77.967

For the experiments, we have used a publicly-available TensorFlow implementation of the BiLSTM-CRF [32] [3] instead of [3]'s Theano implementation since we needed features of this environment (the inference of y_i^* in (7) is performed by an external, compiled function and stored in the computational graph at run time). Each training session was run until convergence of the evaluation loss function (CoNLL) over the validation set or a maximum of 120 epochs. All hyper-parameters are as in [32], and all the digits have been replaced with zeros as pre-processing. The same pre-trained word embeddings used in [3] have been used for training initialization. Table 1 compares the cross-entropy accuracy reported in [3] and that obtained by us using the implementation from [32]. As the table shows, we have obtained a mildly lower accuracy for English, lower accuracies for Spanish and Dutch, and a significantly higher accuracy for German. Such differences at a parity of training loss can be explained with the different software implementations of the BiLSTM-CRF. However, they are not the focus of this paper.

As approaches, we have compared: 1) the conventional cross-entropy training of the BiLSTM-CRF; 2) training with a hinge loss bounding the Hamming loss from above (a straightforward and well-known loss-augmented inference); 3) training with a hinge loss bounding the CoNLL loss based on the proposed algorithm; and 4) training with a mixed hinge loss bounding the CoNLL loss for sentences with $\leq 1\%$ of entity tokens, and the Hamming loss otherwise. The value for the threshold was chosen by running an initial

2. Three anomalously long sentences containing only numerical values in the Spanish training set, of respective length $1,238$, 314, and 261 tokens, and three long sentences containing only mathematical equations in the Dutch training set, of respective length 859, 708, and 454, have been split into shorter sentences of length 155 ± 25.

3. https://github.com/Franck-Dernoncourt/NeuroNER

TABLE 2
The compared training objectives.

Cross Entropy	$l_{CrossEntropy} = -\sum_{i=1}^{N} \log p(y_i \vert x_i)$
Hinge-Hamming	$l_{Hinge-Hamming} = -\sum_{i=1}^{N}[-F(x_i, y_i) + F(x_i, y_i^*) + \Delta_{Hamming}(y_i, y_i^*)]_+$ $y_i^* = \text{argmax}_y\, F(x_i, y) + \Delta_{Hamming}(y_i, y)$
Hinge-CoNLL	$l_{Hinge-CoNLL} = -\sum_{i=1}^{N}[-F(x_i, y_i) + F(x_i, y_i^*) + \Delta_{CoNLL}(y_i, y_i^*)]_+$ $y_i^* = \text{argmax}_y\, F(x_i, y) + \Delta_{CoNLL}(y_i, y)$
Mixed Hinge	$l_{MixedHinge} = \begin{cases} l_{Hinge-CoNLL}, & \text{if } EntityDensity(y_i) \leq th \\ l_{Hinge-Hamming}, & \text{otherwise} \end{cases}$

TABLE 3
Comparison of the CoNLL scores with the different loss functions.

Experiment	English	German	Spanish	Dutch
Cross Entropy	90.412 ± 0.245	82.156 ± 0.331	84.312 ± 0.637	77.967 ± 0.564
Hinge-Hamming	90.406 ± 0.196	82.322 ± 0.471	83.815 ± 0.357	77.835 ± 0.542
Hinge-CoNLL	90.294 ± 0.303	81.289 ± 0.355	83.414 ± 0.425	77.387 ± 0.416
Mixed Hinge	$\mathbf{90.497 \pm 0.149}$	$\mathbf{82.349 \pm 0.318}$	$\mathbf{84.525 \pm 0.276}$	$\mathbf{78.126 \pm 0.417}$

TABLE 4
Comparison of the MUC scores with the different loss functions.

Experiment	English	German	Spanish	Dutch
Cross Entropy	93.375 ± 0.157	85.363 ± 0.304	90.808 ± 0.354	86.201 ± 0.404
Hinge-Hamming	93.393 ± 0.182	85.481 ± 0.465	90.571 ± 0.210	86.276 ± 0.332
Hinge-CoNLL	93.327 ± 0.179	84.475 ± 0.275	90.277 ± 0.248	85.745 ± 0.373
Mixed Hinge	$\mathbf{93.452 \pm 0.094}$	$\mathbf{85.489 \pm 0.299}$	$\mathbf{90.944 \pm 0.187}$	$\mathbf{86.537 \pm 0.252}$

experiment on the English dataset with threshold values between 0.5% and 2%, and selecting the best value based on the accuracy on the validation set. The same value was applied unchanged in all experiments. The percentage of sentences within the 1% threshold is significant, ranging from 25.1% for Spanish to 53.9% for Dutch. All the training objectives are displayed in Table 2.

Performance evaluation has been carried out using four different performance measures: the CoNLL score, the MUC score, an F_1 score for entity segmentation alone, and an F_1 score for entity classification. The CoNLL score has been computed using the standard scoring script[4]. The MUC score, too, has been computed using the official scorer[5]. This score is generally higher than the CoNLL score since the predictions are evaluated separately in terms of class and segmentation [33]. Sections 5.1 and 5.2 report the results for these four performance measures while computational times are reported and discussed in Section 5.3.

In addition to the main set of experiments, we have carried out a comparative experiment with the high-performing ELMo embeddings [10] and the DeLFT implementations of the BiLSTM-CRF [3] and BiLSTM-CNN-CRF [4] over the English Data set. The BiLSTM-CNN-CRF is a hybrid architecture which uses the same outer layers as the BiLSTM-CRF, but encodes the character sequence of each token using a convolutional neural network (CNN). We have included this model in the experiment to test the ability of the mixed hinge loss to produce performance improvements with different models than the standard BiLSTM-CRF. The results are presented in Section 5.4.

4. Publicly available at https://www.clips.uantwerpen.be/conll2002/ner/bin/conlleval.txt.

5. Publicly available at https://catalog.ldc.upenn.edu/docs/LDC2001T02/MUC_scorer3.3/.

5.1 Performance Evaluation with the CoNLL and MUC Scores

Table 3 shows the comparison of the CoNLL scores obtained by training the model with the different loss functions. The scores have been computed over the test set at the epoch with the best dev-set score. To marginalize the effects of the random initialization and the dropout, we have repeated each experiment 10 times and reported the average and the standard deviation. Moreover, since the loss-augmented inference is sensitive to the scale of the loss [5], we have carried out an initial sensitivity analysis over each dataset to find an appropriate scale for the Hamming and CoNLL losses.

Table 3 shows that the scores achieved by the mixed hinge loss have been slightly higher than those of the cross entropy across all four languages (from 0.08 percentage points for English to 0.21 for Spanish). The scores achieved by the hinge-Hamming loss alone have been generally worse than those of the cross entropy (mildly higher for German, but worse for English, Spanish and Dutch). In turn, hinge-Hamming has outperformed hinge-CoNLL on all four languages. This result is surprising to an extent since the hinge-CoNLL objective is closer to the evaluation measure. Aside from possible overfitting, a plausible explanation is that the Hamming loss is a "finer" loss than CoNLL and can provide a greater range of violating labelings in (7). As explained in Section 4.2, the mixed hinge aims to capture the best of both losses, by using the CoNLL loss as the training objective for sentences with sparser entities, where it penalizes errors more severely than the Hamming loss, and the Hamming loss for all others. Its results are a clear improvement over either separate loss, with an average increase of 0.28 points over Hamming and 0.78 over

TABLE 5
Comparison of the segmentation F_1 scores with the different loss functions.

Experiment	English	German	Spanish	Dutch
Cross Entropy	94.837 ± 0.142	85.807 ± 0.328	94.105 ± 0.471	$\mathbf{91.991 \pm 0.410}$
Hinge-Hamming	94.806 ± 0.225	85.925 ± 0.597	94.149 ± 0.290	91.835 ± 0.387
Hinge-CoNLL	94.575 ± 0.152	84.609 ± 0.317	93.813 ± 0.383	91.238 ± 0.508
Mixed Hinge	$\mathbf{94.893 \pm 0.112}$	$\mathbf{86.085 \pm 0.336}$	$\mathbf{94.237 \pm 0.486}$	91.989 ± 0.349

TABLE 6
Comparison of the classification F_1 scores with the different loss functions.

Experiment	English	German	Spanish	Dutch
Cross Entropy	91.487 ± 0.280	84.011 ± 0.344	86.981 ± 0.589	79.500 ± 0.494
Hinge-Hamming	91.699 ± 0.214	$\mathbf{84.493 \pm 0.532}$	87.001 ± 0.395	80.012 ± 0.428
Hinge-CoNLL	91.618 ± 0.359	83.468 ± 0.389	86.190 ± 0.475	79.297 ± 0.500
Mixed Hinge	$\mathbf{91.748 \pm 0.212}$	84.184 ± 0.264	$\mathbf{87.296 \pm 0.295}$	$\mathbf{80.121 \pm 0.439}$

CoNLL, with a t-test one-tailed p-value < 0.01 in the case of German and Spanish.

In turn, Table 4 shows the MUC scores obtained with the different loss functions. Again, the highest scores have been achieved by the mixed hinge, with an improvement over the cross entropy ranging from 0.08 percentage points for English to 0.33 for Dutch. As expected, the absolute scores for MUC are higher than the CoNLL scores, but the relative rankings of all the training losses are comparable. In particular, the MUC scores for Dutch are much higher than the corresponding CoNLL scores of Table 3, showing that for this language it is harder to attain both entity segmentation and classification accuracy.

5.2 Performance Evaluation with the Entity Segmentation and Entity Classification F_1 Scores

To compute the entity segmentation F_1 score, we have taken all the entity tokens (both ground truth and predicted) and changed their class to a single, notional class (say, X). In this way, the classes are treated only as B-X, I-X and O, and running the CoNLL scorer assesses the segmentation quality alone. The entity classification F_1 score has been computed by instead ignoring all the B- and I- prefixes, using the CoNLL scorer with the provided '-r' option. The separate evaluation of the segmentation and classification performance is likely to shed more light on the compared models.

Table 5 reports the segmentation F_1 scores obtained with the different loss functions. On this measure, the mixed hinge loss has achieved higher scores for three languages while the cross entropy has achieved a notionally higher score for Dutch. For English and German, the segmentation scores are similar to the MUC scores of Table 4. However, for Spanish and Dutch the segmentation scores are much higher than the corresponding MUC scores, showing that classification is relatively harder than segmentation for these two languages.

Eventually, Table 6 shows the classification F_1 scores obtained with the different loss functions. Also on this measure, the highest scores have been achieved by the mixed hinge loss in most cases, with an improvement over the cross entropy ranging from 0.18 percentage points for English to 0.62 for Dutch (one-tailed p-value < 0.05), with an average of 0.32 percentage points across all languages.

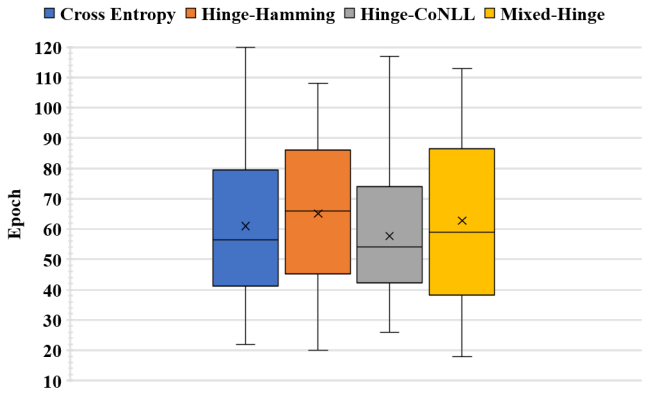

Fig. 3. Training epoch of maximum validation accuracy for the different loss functions.

The relative rankings for the other losses are again similar, but the cross entropy has achieved a more limited performance over Dutch.

5.3 Computational Times

As computational times are concerned, training with the cross entropy has proved the fastest, followed by the hinge-Hamming and the hinge-CoNLL, respectively. For instance, training one epoch of the English data set on a 3.4 GHz Intel Xeon with 32 GB of RAM has taken 177 s with the cross entropy, 279 s with the hinge-Hamming, 464 s with the hinge-CoNLL, and 297 s with the mixed hinge. Figure 3 shows the epoch at which each training loss has reached the maximum validation accuracy over all languages and runs. This figure shows that the rate of convergence is comparable for all training losses. Overall, the total training times have proved manageable for all losses.

5.4 A Comparative Experiment Using the ELMo Contextualized Embeddings

In addition to the above experiments, we have performed a comparative experiment over the English dataset using the high-performing ELMo contextualized embeddings [10]. For this experiment, we have used the DeLFT BiLSTM-CRF and BiLSTM-CNN-CRF with their default word em-

TABLE 7
Comparison of the CoNLL scores over the English dataset with the different loss functions and ELMo word embeddings using two models.

Experiment	BiLSTM-CRF	BiLSTM-CNN-CRF
Cross Entropy	92.076 ± 0.087	92.263 ± 0.098
Hinge-Hamming	92.026 ± 0.220	92.400 ± 0.115
Hinge-CoNLL	90.873 ± 0.381	91.390 ± 0.243
Mixed Hinge	$\mathbf{92.206 \pm 0.136}$	$\mathbf{92.626 \pm 0.187}$

beddings (a concatenation of GLoVe-300d [6] and ELMo-1024d [7]). Table 7 shows the CoNLL scores obtained with the different loss functions as average of 3 independent runs. These results show that the proposed mixed hinge loss has been able to achieve, again, mildly higher scores than the cross entropy with both the BiLSTM-CRF (0.13 percentage points) and the BiLSTM-CNN-CRF (0.36 percentage points), and higher than those of the Hamming and CoNLL losses in isolation. In addition, all the improvements of the mixed hinge loss over the other losses have a one-tailed p-value < 0.02 for the BiLSTM-CRF and < 0.01 for the BiLSTM-CNN-CRF, and can therefore be regarded as statistically significant.

Finally, Table 8 shows examples of outputs obtained on the CoNLL 2003 English test set using the BiLSTM-CNN-CRF and the different training losses. For each loss, we have used the predictions from the run with the highest CoNLL score on the test set. The example in the top rows shows a sentence where the mixed hinge loss has been able to provide the correct prediction, while all other losses have made a combination of classification and segmentation errors. The example in the middle rows shows a challenging sentence with four close-by entities. In this case, all losses have only been able to recognize the first correctly. However, the predictions from the mixed hinge and hinge-CoNLL losses show a "more graceful" degradation. Eventually, the bottom rows show another example where the mixed hinge loss has been able to make a correct prediction while the cross entropy has generated a false positive and the other hinge losses have made a classification error. All these examples also show that, despite its intuitive definition, named-entity recognition is a far-from-trivial task.

6 Conclusion

In this paper, we have presented the BiLSTM-SSVM, a training approach for the BiLSTM-CRF based on a hinge loss minimization. In the approach, the hinge loss is used as an upper bound for three evaluation losses, namely Hamming, CoNLL and a combination of the two. The required loss-augmented inference is challenging in the case of a non-decomposable loss such as CoNLL, and, for this reason, in this paper we have proposed an articulated dynamic programming algorithm that can perform the loss-augmented inference for the CoNLL loss and any other loss similarly based on entity-level error counting. Since the CoNLL loss is of the F_1 type, we have also argued that it may be a promising training objective for sentences with relatively sparse entities. For this reason, we have proposed a training objective that bounds the CoNLL loss for sentences with low entity density, and the Hamming loss otherwise ("mixed hinge"). Experiments conducted over four benchmark languages (English, German, Spanish and Dutch) have shown that training with the mixed hinge loss has achieved slightly higher accuracies than with the cross entropy for all languages. These results suggest that training with objectives closer to the evaluation measures can be an effective strategy, and that using different losses for sentences with different sufficient statistics should be explored further. As such, we plan to soon extend our investigation to a variety of other tasks, losses and architectures.

6. http://nlp.stanford.edu/data/glove.840B.300d.zip
7. https://allennlp.org/elmo

Acknowledgments

This research has been funded by the Capital Markets Cooperative Research Centre in Australia. The authors are specially indebted to Ben Hachey for his invaluable advice on this paper.

References

[1] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," *Linguisticae Investigationes*, vol. 30, no. 1, pp. 3–26, 2007.
[2] Z. Huang, W. Xu, and K. Yu, "Bidirectional LSTM-CRF models for sequence tagging," *CoRR*, vol. abs/1508.01991, 2015.
[3] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, C. D. Lebret, and R. Collobert, "Neural architectures for named entity recognition," in *Proceedings of the 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 260–270.
[4] X. Ma and E. Hovy, "End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, 2016, pp. 1064–1074.
[5] T. Joachims, "A support vector method for multivariate performance measures," in *Proceedings of the 22nd International Conference on Machine Learning*, 2005, pp. 377–384.
[6] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, pp. 2493–2537, 2011.
[7] G. Mesnil, X. He, L. Deng, and Y. Bengio, "Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding," in *Proceedings of the 14th Annual Conference of the International Speech Communication Association*, 2013, pp. 3771–3775.
[8] G. Mesnil, Y. Dauphin, K. Yao, Y. Bengio, L. Deng, D. Hakkani-Tur, X. He, L. Heck, G. Tur, D. Yu, and G. Zweig, "Using recurrent neural networks for slot filling in spoken language understanding," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 3, pp. 530–539, 2015.
[9] M. E. Peters, W. Ammar, C. Bhagavatula, and R. Power, "Semi-supervised sequence tagging with bidirectional language models," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 2017, pp. 1756–1765.
[10] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: HLT*, 2018, pp. 2227–2237.
[11] L. Liu, J. Shang, F. F. Xu, X. Ren, H. Gui, J. Peng, and J. Han, "Empower Sequence Labeling with Task-Aware Neural Language Model," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, 2018, pp. 5253–5260.
[12] M. Rei, "Semi-supervised multitask learning for sequence labeling," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 2017, pp. 2121–2130.
[13] S.-X. Zhang, R. Zhao, C. Liu, J. Li, and Y. Gong, "Recurrent support vector machines for speech recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2016, pp. 5885–5889.

TABLE 8
Examples of NER outputs with the different training losses.

Token	Ground Truth	Cross Entropy	Hinge-Hamming	Hinge-CoNLL	Mixed Hinge
BADMINTON	O	O	O	O	O
-	O	O	O	O	O
WORLD	B-MISC	O	O	B-LOC	B-MISC
GRAND	I-MISC	B-MISC	B-MISC	B-MISC	I-MISC
PRIX	I-MISC	I-MISC	I-MISC	I-MISC	I-MISC
RESULTS	O	O	O	O	O
.	O	O	O	O	O
NCAA	B-ORG	B-ORG	B-ORG	B-ORG	B-ORG
AMERICAN	O	B-MISC	B-MISC	B-MISC	B-MISC
FOOTBALL-OHIO	B-MISC	O	O	I-MISC	I-MISC
STATE	I-MISC	O	O	O	O
'S	O	O	O	O	O
PACE	B-PER	O	O	O	O
FIRST	O	O	O	O	O
REPEAT	O	O	O	O	O
LOMBARDI	B-MISC	O	O	B-PER	B-PER
AWARD	I-MISC	O	O	O	O
WINNER	O	O	O	O	O
.	O	O	O	O	O
GOLF	O	B-LOC	O	O	O
-	O	O	O	O	O
ZIMBABWE	B-MISC	B-MISC	B-LOC	B-LOC	B-MISC
OPEN	I-MISC	I-MISC	I-MISC	I-MISC	I-MISC
SECOND	O	O	O	O	O
ROUND	O	O	O	O	O
SCORES	O	O	O	O	O
.	O	O	O	O	O

[14] Y. Shi, K. Yao, H. Chen, D. Yu, Y. Pan, and M. Hwang, "Recurrent support vector machines for slot tagging in spoken language understanding," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 393–399.

[15] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, "Large margin methods for structured and interdependent output variables," *Journal of Machine Learning Research*, vol. 6, pp. 1453–1484, 2005.

[16] Y. Adi, J. Keshet, E. Cibelli, and M. Goldrick, "Sequence segmentation using joint RNN and structured prediction models," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2017, pp. 2422–2426.

[17] J. Suzuki, E. McDermott, and H. Isozaki, "Training conditional random fields with multivariate evaluation measures," in *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, 2006, pp. 217–224.

[18] N. Rosenfeld, O. Meshi, D. Tarlow, and A. Globerson, "Learning structured models with the AUC loss and its generalizations," in *Proceedings of the 17th International Conference on Artificial Intelligence and Statistics*, 2014, pp. 841–849.

[19] G. Haffari, A. Nagesh, and G. Ramakrishnan, "Optimizing multivariate performance measures for learning relation extraction models," in *Proceedings of the 2015 Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2015, pp. 892–900.

[20] G. Zhang, M. Piccardi, and E. Z. Borzeshi, "Sequential labeling with structural SVM under non-decomposable losses," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, pp. 4177–4188, 2017.

[21] D. McAllester and J. Keshet, "Generalization bounds and consistency for latent structural probit and ramp loss," in *Proceedings of the 24th International Conference on Neural Information Processing Systems*, 2011, pp. 2205–2212.

[22] K. Gimpel and N. A. Smith, "Structured ramp loss minimization for machine translation," in *Proceedings of the 2012 Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2012, pp. 221–231.

[23] X. Huang, L. Shi, and J. A. Suykens, "Ramp loss linear programming support vector machine," *Journal of Machine Learning Research*, vol. 15, pp. 2185–2211, 2014.

[24] D. Karmon and J. Keshet, "Risk minimization in structured prediction using orbit loss," *CoRR*, vol. abs/1512.02033, 2015.

[25] D. McAllester, T. Hazan, and J. Keshet, "Direct loss minimization for structured prediction," in *Proceedings of the 23rd International Conference on Neural Information Processing Systems*, 2010, pp. 1594–1602.

[26] Y. Song, A. G. Schwing, R. S. Zemel, and R. Urtasun, "Training deep neural networks via direct loss minimization," in *Proceedings of the 33rd International Conference on Machine Learning*, 2016, pp. 2169–2177.

[27] C. Sutton and A. McCallum, "An introduction to conditional random fields," *Foundations and Trends in Machine Learning*, vol. 4, no. 4, pp. 267–373, 2012.

[28] Y. Adi and J. Keshet, "StructED: Risk minimization in structured prediction," *Journal of Machine Learning Research*, vol. 17, no. 64, pp. 1–5, 2016.

[29] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer-Verlag New York, Inc., 1995.

[30] E. F. Tjong Kim Sang and F. De Meulder, "Introduction to the conll-2003 shared task: Language-independent named entity recognition," in *Proceedings of the 7th Conference on Natural Language Learning*, vol. 4, 2003, pp. 142–147.

[31] E. F. Tjong Kim Sang, "Introduction to the conll-2002 shared task: Language-independent named entity recognition," in *Proceedings of the 6th Conference on Natural Language Learning*, vol. 20, 2002, pp. 1–4.

[32] F. Dernoncourt, J. Y. Lee, and P. Szolovits, "NeuroNER: An easy-to-use program for named-entity recognition based on neural networks," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2017, pp. 97–102.

[33] N. Chinchor, "MUC-7 named entity task definition dry run version, version 3.5, 17 september 1997," in *Proceedings of the Seventh Message Understanding Conference*. Fairfax, Virginia: Morgan Kaufmann Publishers, Inc., 1998.

Hanieh Poostchi (S'11,17,19) received an M.Sc. in Artificial Intelligence from Ferdowsi University of Mashhad, Iran in 2012. She was associated with the Department of Computer Science at Aalto University, Espoo, Uusimaa, Finland from 2013 to 2015. Recently, she completed a Ph.D. degree at University of Technology Sydney, Ultimo, NSW, Australia. Her current research interests include training deep recurrent neural networks (RNNs) with structural losses for low-level natural language processing sequential labeling tasks.

Massimo Piccardi (SM'05) received an M.Eng. and Ph.D. degrees from the University of Bologna, Bologna, Italy, in 1991 and 1995, respectively. He is currently a Full Professor of computer systems with University of Technology Sydney, Ultimo, NSW, Australia. His research interests include natural language processing, computer vision and pattern recognition and he has co-authored over 150 papers in these areas.

Dr. Piccardi is a Senior Member of the IEEE, a member of its Computer and Systems, Man, and Cybernetics Societies, and a member of the International Association for Pattern Recognition. He serves as an Associate Editor for the IEEE Transactions on Big Data and journal Computer Vision and Image Understanding.