Linear Co-occurrence Rate Networks (L-CRNs) for Sequence Labeling

Zhemin Zhu(✉), Djoerd Hiemstra, and Peter Apers

Electrical Engineering, Mathematics and Computer Science, University of Twente, Drienerlolaan 5, 7500 AE Enschede, The Netherlands
{z.zhu,d.hiemstra,p.m.g.apers}@utwente.nl

Abstract. Sequence labeling has wide applications in natural language processing and speech processing. Popular sequence labeling models suffer from some known problems. Hidden Markov models (HMMs) are generative models and they cannot encode transition features; Conditional Markov models (CMMs) suffer from the label bias problem; And training of conditional random fields (CRFs) can be expensive. In this paper, we propose Linear Co-occurrence Rate Networks (L-CRNs) for sequence labeling which avoid the mentioned problems with existing models. The factors of L-CRNs can be locally normalized and trained separately, which leads to a simple and efficient training method. Experimental results on real-world natural language processing data sets show that L-CRNs reduce the training time by orders of magnitudes while achieve very competitive results to CRFs.

Keywords: Sequence labeling · Co-occurrence rate · HMMs · CRFs

1 Introduction

Sequence labeling is a sub-task of structured prediction. A wide range of fundamental applications in natural language processing and speech processing can be formulated as sequence labeling models, such as named entity recognition, part-of-speech tagging and speech recognition. A common nature of these applications is that these applications desire a sequence of labels as output rather than a single label. This makes sequence labeling stand out from the typical supervised classification tasks which normally predict a single label as output. Here we give a simplified example of named entity recognition (NER) to illustrate the typical scenario of sequence labeling. Given a sentence, which consists of a sequence of words, NER systems assign each word of the sentence a label. These labels indicate the types of named entities, such as location (LOC), person (PER), organization (ORG), or out of any named entity (O).

$[\texttt{Jimmy}]_{\texttt{PER}}$ $[\texttt{de}]_{\texttt{PER}}$ $[\texttt{Graff}]_{\texttt{PER}}$ $[\texttt{is}]_{\texttt{O}}$ $[\texttt{a}]_{\texttt{O}}$ $[\texttt{member}]_{\texttt{O}}$ $[\texttt{of}]_{\texttt{O}}$ $[\texttt{the}]_{\texttt{O}}$ $[\texttt{Dutch}]_{\texttt{ORG}}$ $[\texttt{National}]_{\texttt{ORG}}$ $[\texttt{Research}]_{\texttt{ORG}}$ $[\texttt{School}]_{\texttt{ORG}}$ $[\texttt{for}]_{\texttt{ORG}}$ $[\texttt{Knowledge}]_{\texttt{ORG}}$ $[\texttt{Systems}]_{\texttt{ORG}}$.

Our C++ implementation of L-CRNs and the datasets used in this paper can be found at https://github.com/zheminzhu/Co-occurrence-Rate-Networks.

© Springer International Publishing Switzerland 2014
L. Besacier et al. (Eds.): SLSP 2014, LNAI 8791, pp. 185–196, 2014.
DOI: 10.1007/978-3-319-11397-5_14

The words in a sentence are observations. From this example, we can see that intuitively two kinds of information can affect the prediction of the label at current position:

1. Label dependence. Adjacent labels can affect prediction of the current label. For example, if the adjacent labels are `ORG`, the current label is more likely to be `ORG`.
2. Observation evidence. The current word observed can affect the current label. For example, the word `Dutch` is more likely to be `ORG` than the word `is`.

Accordingly, a sequence labeling model should do the following three tasks well.

– Task 1. Modeling label dependence.
– Task 2. Modeling observation evidence.
– Task 3. Combining these two parts to obtain results.

Task 3 has been paid less attention. The two parts should be given relative weights properly when we combine them. As we will discuss, failure in doing this will lead to a subtle problem called the label bias problem [11], in which label dependence is given too much weight and observation evidence is underestimated or even ignored.

Due to its wide applications, sequence labeling has been heavily studied for a long history. There exist a rich set of popular models for sequence labeling, such as hidden Markov models (HMMs) [14], conditional Markov models (CMMs) [13] and conditional random fields (CRFs) [11].[1] The general idea under all of these models is factorization. That is to decompose a high-dimensional joint probability into a product of small factors based on some conditional independence assumptions. A model is characterized by its factorization. Hence we can see the pros and cons of a model from its factorization.

1.1 Hidden Markov Models (HMMs)

Figure 1 shows a first order HMM. $S = [s_1, s_2, \ldots, s_n]$ is the label sequence and $O = [o_1, o_2, \ldots, o_n]$ is the observation sequence. In the NER example, S is the sequence of NER labels and O is the sequence of words. HMMs are directed

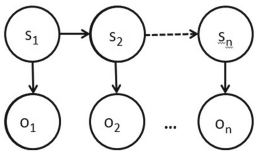

Fig. 1. Hidden Markov models

[1] Another popular model is structured (structural) SVM [1] which essentially applies factorization to kernels. Due to its lack of a direct probabilistic interpretation, we leave it for future work.

and generative models. Hence HMMs can also be considered as a special Bayesian network [6]. HMMs factorize a joint probability as follows:

$$p(S,O) \approx p(s_1)\prod_{i=1}^{n} p(o_i|s_i)\prod_{j=1}^{n-1} p(s_{j+1}|s_j). \tag{1}$$

The factors of HMM are probabilities which can be locally normalized. There are two known drawbacks with HMMs [4] which can be observed from Eq. 1. The first drawback is the label transition probabilities $p(s_{j+1}|s_j)$ in HMMs are not conditioned by observations. That is, HMMs use the universal transition probabilities $p(s_{j+1}|s_j)$ without respect to observations. Hence we cannot use observation evidence to help predicting label transition probabilities. Transition features extracted from observation evidence contain valuable information. The second drawback is called mismatch problem. In training stage, HMMs optimize a joint probability $p(S,O)$. But in decoding stage, we search for a sequence of labels which maximizes a conditional probability $p(S|O)$. Klein et al. [9] show that the mismatch problem can reduce accuracy.

To avoid the mismatch problem, we need to directly factorize the conditional probability $p(S|O)$. And in order to encode the transition features, we can set observation evidence to conditions of the transition factors. Conditional Markov models just implement these ideas.

1.2 Conditional Markov Models (CMMs)

Figure 2 shows a CMM. Maximum entropy markov models (MEMMs) [13] are typical CMMs which train the model using a maximum entropy framework, which was later shown to be equivalent to maximum likelihood estimation. CMMs are discriminative models which factorize a conditional probability:

$$p(S|O) = p(s_1|O)\prod_{i=1}^{n-1} p(s_{i+1}|s_i,O). \tag{2}$$

CMMs avoid the mismatch problem of HMMs because they directly factorize $P(S|O)$. And probabilities $p(s_{i+1}|s_i,O)$ predicting the next label are conditioned by previous label s_i together with the observation O. In this way, the transition features can be encoded into CMMs. Hence the first drawback of HMMs is avoided. But this causes a new problem. By putting the previous label s_i

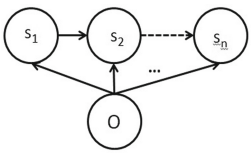

Fig. 2. Conditional Markov models

and observations O together in the condition leads to the label bias problem (LBP). Intuitively, this is because the label dependence (given by s_i) and observation evidence (given by O) are mixed together in one factor. One of them may dominate the factor when its distribution is of low entropy, while the other is underestimated or even ignored. An extreme case is when s_i has only one possible out-going transition s_{i+1}[2], then $p(s_{i+1}|s_i, O)$ is always equal to 1 no matter what O is. That is the observation evidence O is ignored and the label dependence dominates the results. Hence CMMs do not perform the Task 3 perfectly. See [11,12,19] for more examples and discussions.

To avoid the label bias problem, we need to guarantee that the observation evidence can always be used in prediction[3]. This can be done by decoupling the label dependence and observation evidence into different factors, such that none of them can dominate the other. Conditional random fields implement this.

1.3 Conditional Random Fields (CRFs)

Figure 3 shows a linear-chain conditional random field. CRFs [11] are discriminative and undirected graphical models. The factorization for undirected models is based on the Hemmersley-Clifford Theorem [7] which implies a linear-chain CRF can be factorized as follows:

$$p(S|O) = \frac{1}{Z_O} \prod_{i=1}^{n-1} \psi(s_i, s_{i+1}, O) \prod_{j=1}^{n} \phi(s_j, O),$$

Z_O is a global normalization constant, also called partition function, which ensures $\sum_S p(S|O) = 1$. ψ and ϕ are non-negative factors defined over pairwise and unary cliques. The factors of local models, such as HMMs and CMMs, are probabilities. These factors can be locally normalized. By contrast, CRFs are globally normalized models. The factors of CRFs, ψ and ϕ, have no probabilistic interpretations[4] and cannot be locally normalized.

CRFs model the conditional probability $P(S|O)$. Hence they avoid the mismatch problem of HMMs. Also the bigram factors $\psi(s_i, s_{i+1}, O)$ modeling label

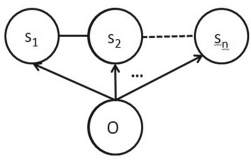

Fig. 3. Conditional random fields

[2] In this extreme case, the entropy of $p(s_{i+1}|s_i)$ is the lowest: 0.

[3] HMMs do not suffer from the label bias problem, because the factors $p(o_i|s_i)$ in Eq. 1 guarantee that the observation evidence is always used.

[4] Sometimes they are intuitively explained as the compatibility of the nodes in cliques. But the notion compatibility has no mathematical definition.

dependence include the observations. Hence the transition features can be encoded into CRFs. Furthermore, CRFs decouple the label dependence (modeled by $\psi(s_i, s_{i+1}, O)$) and observation evidence (modeled by $\phi(s_j, O)$) into different factors. This guarantees that none of them can dominate the other. Obviously, unigram factors $\phi(s_j, O)$ guarantee that O is always used for prediction. Therefore the label bias problem is avoided. Nevertheless, training of CRFs can be very expensive [4,16]. This is because we need to re-calculate the global partition function Z_O for each instance in each optimization iteration.

In this paper, we propose a model called Linear Co-occurrence Rate Networks (L-CRN) for sequence labeling. L-CRNs avoid the problems mentioned above. More specifically, L-CRNs model a conditional probability. Hence they avoid the mismatch problem of HMMs. The label dependence is modeled by the quantity called Co-occurrence Rate (CR), which is conditioned by observations. In this way, transition features can be easily encoded into L-CRNs. Furthermore, in the factorization of L-CRNs, the label dependence and observation evidence are decoupled into difference factors. Thus none of them can dominate the other. The label bias problem is naturally avoided. Finally, L-CRNs are local models. The factors of L-CRNs can be locally normalized and trained separately. This leads to a very efficient maximum likelihood training method. Experiments on real-world datasets show that L-CRNs reduce the training time by orders of magnitudes and achieve very competitive, even slightly better, results to CRFs.

The rest of this paper is organized as follows. In Sect. 2, we present the co-occurrence rate networks and show that this model avoids the problems mentioned above. Section 3 describes the details of learning and decoding. Experiments are reported in Sect. 4. Conclusions follow in the last section.

2 Linear Co-occurrence Rate Networks (L-CRN)

Firstly, we define a quantity which is called Co-occurrence Rate (CR) as follows:

$$\mathrm{CR}(X_1; X_2; \ldots; X_n) := \frac{p(X_1, \ldots, X_n)}{p(X_1) \ldots p(X_n)}.$$

For convenience, CR with a single variable is defined to be 1. Intuitively, if $\mathrm{CR} > 1$, the events are *attractive*; If $\mathrm{CR} = 1$, the events are *independent*; And if $\mathrm{CR} < 1$, the events are *repulsive*. CR is the exponential function of pointwise mutual information [3], and also related to Copulas [21]. Furthermore, we distinguish the following two notations:

$$\mathrm{CR}(X_1; X_2; X_3) := \frac{p(X_1, X_2, X_3)}{p(X_1)p(X_2)p(X_3)}, \quad \mathrm{CR}(X_1X_2; X_3) := \frac{p(X_1, X_2, X_3)}{p(X_1, X_2)p(X_3)}.$$

The first one is the CR between three variables. By contrast, the second one is the CR between a joint variable (X_1X_2) and a single variable (X_3). More comprehensive description of CR can be found in [18,20]. The factorization of L-CRN consists of steps:

1. Decouple the conditional probability into two parts:

$$p(s_1, \ldots, s_n|O) = \mathrm{CR}(s_1; s_2; \ldots; s_n|O) \prod_{i=1}^{n} p(s_i|O),$$

where

$$\mathrm{CR}(s_1; s_2; \ldots; s_n|O) := \frac{p(s_1, \ldots, s_n|O)}{\prod_{i=1}^{n} p(s_i|O)}.$$

This step seems trivial. But this is the key to avoid the label bias problem. The conditional probability is decoupled into two parts: $\mathrm{CR}(s_1; s_2; \ldots; s_n|O)$ models label dependence and $\prod_{i=1}^{n} p(s_i|O)$ models observation evidence. So none of them can dominate the other. $\prod_{i=1}^{n} p(s_i|O)$ guarantees that the observation O is always used for prediction. Hence the label bias problem is naturally avoided. We show this experimentally in [19].

2. Further factorize the joint CR into a product of smaller CR*s* according to Theorems 1 and 2. See Sect. 6.2 for their proofs.

Theorem 1 (Partition Operation). $\mathrm{CR}(X_1; \ldots; X_j; X_{j+1}; \ldots; X_n) = \mathrm{CR}(X_1; ..; X_j)\mathrm{CR}(X_{j+1}; ..; X_n)\mathrm{CR}(X_1..X_j; X_{j+1}..X_n)$

Theorem 2 (Reduce Operation). *If* $X \perp\!\!\!\perp Y \mid Z$, *then* $\mathrm{CR}(X; YZ) = \mathrm{CR}(X; Z)$.

$X \perp\!\!\!\perp Y \mid Z$ means X is independent of Y conditioned by Z. Putting two steps together, the factorization of L-CRN is obtained as follow:

$$\begin{aligned}
p(s_1, s_2, \ldots, s_n \mid O) &= \mathrm{CR}(s_1; \ldots; s_n \mid O) \prod_{i=1}^{n} p(s_i \mid O) \\
&= \mathrm{CR}(s_1|O)\mathrm{CR}(s_2; \ldots; s_n|O)\mathrm{CR}(s_1; s_2 \ldots s_n|O) \prod_{i=1}^{n} p(s_i|O) \\
&= \mathrm{CR}(s_2; \ldots; s_n \mid O)\mathrm{CR}(s_1; s_2 \mid O) \prod_{i=1}^{n} p(s_i \mid O) \\
&\ldots \\
&= \prod_{j=1}^{n-1} \mathrm{CR}(s_j; s_{j+1} \mid O) \prod_{i=1}^{n} p(s_i \mid O).
\end{aligned}$$

The second equation is obtained by partitioning s_1 out. We obtain the third equation from the second by $\mathrm{CR}(s_1 \mid O) = 1$ and applying the reduce operation to the factor $\mathrm{CR}(s_1; s_2 \ldots s_n|O)$ since $s_1 \perp\!\!\!\perp s_3 \ldots s_n | s_2$. By repeating this process, we can get the final factorization. Hence we obtain a L-CRN factorization on a chain graph as follows:

$$p(s_1, s_2, \ldots, s_n|O) = \prod_{j=1}^{n-1} \mathrm{CR}(s_j; s_{j+1} \mid O) \prod_{i=1}^{n} p(s_i \mid O). \tag{3}$$

From this factorization, we can see the following facts. L-CRNs model a conditional probability. Hence they avoid the mismatch problem of HMMs. The label dependence of L-CRNs is modeled by $\prod_{j=1}^{n-1} \mathrm{CR}(s_j; s_{j+1}|O)$, which is conditioned by observations. Thus transition features can be easily encoded into L-CRNs. Furthermore, $\prod_{i=1}^{n} p(s_i \mid O)$ guarantee the observation is always used for prediction. Therefore the label bias problem is avoided. Finally, L-CRNs are local models. The factors of L-CRNs can be locally normalized and hence separately trained. This leads to a very simple and efficient maximum likelihood training as described in the next section. [2] shows local models can outperform globally normalized models on some NLP tasks. CR factorization can be extended to arbitrary graphs (Sect. 6.2 of [18]).

3 Learning and Decoding

Since the factors of L-CRNs can be normalized locally and trained separately, the learning of L-CRNs becomes very simple and efficient. It is no more than training a set of regression models for each factor in training stage, and combining them together to find a maximum sequence probability in decoding stage. Accroding to Eq. 3, there are two kinds of factors to be trained: unigram factors $p(s|O)$ and bigram factors $\mathrm{CR}(s; s'|O)$. We describe the details as follows. See Sect. 6.1 for the justification of this training method. In fact, this is the maximum likelihood estimation of Eq. 3.

3.1 Learning Unigram Factor $p(s|O)$

For the factors $p(s|O)$ in Eq. 3, where s is a label and O is an observation. As described in Sect. 6.1, its MLE is just the relative frequency $\hat{p}(s|O) = \frac{\#(s,O)}{\sum_s \#(s,O)}$, where $\#(s, O)$ is the number of times (s, O) appears in the training dataset. This relative frequency can be easily obtained from the training dataset by counting. Let $g_1(O), g_2(O), \ldots, g_n(O)$ be feature functions of O, which are called unigram features with respect to the unigram label s. For each label s in the label space, we train a regression model ϕ_s:

$$\hat{p}(s|O) = \phi_s : (g_1(O), g_2(O), \ldots, g_n(O)) \mapsto \frac{\#(s, O)}{\sum_s \#(s, O)}.$$

In decoding, for an observation O, we use $\phi_s(g_1(O), \ldots, g_n(O))$ as the estimation of $p(s|O)$. If $g_1(O), g_2(O), \ldots, g_n(O)$ has been seen in the training dataset, we just use $\frac{\#(s,O)}{\sum_s \#(s,O)}$ as the estimation of $p(s|O)$. Because this is the MLE of $p(s|O)$ (see Sect. 6.1). Otherwise, ϕ_s is used.

3.2 Learning Bigram Factor CR($s; s'|O$)

Similarly, we train regression models $\psi_{s,s'}$ separately for each bigram label s, s' for predicting :

$$\hat{\mathrm{CR}}(s; s'|O) = \psi_{s,s'} : (h_1(O), h_2(O), \ldots, h_m(O)) \mapsto \frac{\#(s, s', O)}{\sum_{s,s'} \#(s, s', O)} \Big/ \frac{\#(s, O)}{\sum_{s} \#(s, O)} \Big/ \frac{\#(s', O)}{\sum_{s'} \#(s', O)}.$$

$h_1(O), h_2(O), \ldots, h_m(O)$ are bigram features extracted from O. Similarly, in decoding, $\psi_{s,s'}(h_1(O), h_2(O), \ldots, h_m(O))$ are used as the prediction of $\mathrm{CR}(s; s'|O)$. If $h_1(O), h_2(O), \ldots, h_m(O)$ has been observed in the training dataset, we directly use the empirical value. Otherwise, we use $\psi_{s,s'}$.

We use the traditional Viterbi algorithm for selecting the label sequence with maximum probability in decoding stage.

3.3 Support Vector Regression

There exist a rich set of regression models which may be used for modeling ϕ_s and $\psi_{s,s'}$. In this paper, we adopt the support vector regression (SVR) [15] for modeling ϕ_s and $\psi_{s,s'}$ discussed above. SVR is linear in the high dimensional transformed space and tolerant to low error points with small residuals. Such tolerance seems to fit the natural language processing applications well, in which the input and final output are normally categorical. In text classification, large-margin methods achieve very good results [8]. And there are good implementations of SVR which can handle very large number of instances and features efficiently. These reasons lead us to prefer SVR. In future, we will try other regression models. To avoid endowing unwanted metric and order structures to a single categorical variable, we use the dummy coding as the representation of categorical input variables.

4 Experiments

In this section, we compare L-CRN with CRFs[5]. We adopt CRF++ version 0.58 [10] as the implementation for CRFs and LIBLINEAR version 1.94 [5] for linear SVR in L-CRNs. We set the configurations of LIBLINEAR as L2-regularization L2-loss support vector regression (solving dual). For a fair comparison, we always use a single thread for training[6]. We apply CRFs and L-CRNs to an important natural language processing application: named entity recognition (NER).

[5] [11,12] show superiority of CRFs over other models. Hence it is reasonable to compare with CRFs.

[6] L-CRNs can be easily parallellized. Obviously, each regression model can be trained parallely with others.

4.1 Named Entity Recognition

The English part of the CoNLL-2003 NER dataset[7] [17] is used for our NER experiment. There are three data files in this dataset: ner.train, ner.testa and ner.testb. The first one is designed for training and the last two are used for testing. The size of the label space is 8. These three files include 14987, 3466, 3684 sentences and 204567, 51578, 46666 words respectively. We use the same orthographic features as those used by [11]: "whether a spelling begins with a number or upper case letter, whether it contains a hyphen, and whether it ends in one of the following suffixes: −ing, −ogy, −ed, −s, −ly, −ion, −tion, −ity, −ies". Additionally, we use the chunk tags and POS tags provided together with the CoNLL dataset.

Table 1 gives the time taken by CRF and L-CRN. We can see L-CRN reduces the training time significantly.

Tables 2 and 3 show the quality metrics achieved by CRF and L-CRN on ner.testa and ner.testb, respectively. The first three columns show the per-word accuracies (%) on all, known and unknown words[8]. On all and known words, L-CRN consistently outperforms CRF slightly. As described in Sect. 3, L-CRN can directly use empirical values for known word prediction. This may be considered as an advantage of L-CRN. On unknown words, CRF performs better on ner.testa, but L-CRN performs slightly better on ner.testb. The last three columns give the precision, recall and F1 metrics. These metrics were evaluated using the standard CoNLL evaluation tool[9]. CRF obtains better results in precision. L-CRN obtains better results in recall and F1.

Table 1. Training time (seconds) on NER

CRF	L-CRN
1,666	112

Table 2. Metrics on ner.testa

	All	Known	Unknown	Precision	Recall	F1
CRF	97.00	98.27	85.42	84.66	82.31	83.47
L-CRN	97.44	98.80	85.05	84.21	84.45	84.33

Table 3. Metrics on ner.testb

	All	Known	Unknown	Precision	Recall	F1
CRF	95.00	97.46	80.32	75.61	74.70	75.15
L-CRN	95.55	98.06	80.62	75.78	76.43	76.10

[7] http://www.cnts.ua.ac.be/conll2003/ner/

[8] Known words are the words that appear in the training data. Unknown words are the words that have not been seen in the training data. All words include both.

[9] http://www.cnts.ua.ac.be/conll2000/chunking/conlleval.txt

5 Conclusions

We propose the linear co-occurrence rate networks (L-CRN) for sequence labeling. This model avoids problems of the existing models, such as mismatching problems and the label bias problem. The transition features can be encoded into L-CRN. Furthermore, the factors of L-CRN can be normalized locally and trained independently, which leads to very efficient training. In this paper, we use support vector regression as the regression models of factors in L-CRN. Experimental results show L-CRNs reduce the training time by orders of magnitudes and achieve very competitive results to CRFs on real-world NLP data.

Acknowledgments. We thank SLSP 2014 reviewers for their comments. This work has been supported by the Dutch national program COMMIT/.

6 Appendix

6.1 Closed-Form MLE Training of L-CRN

We maximize the log likelihood of Eq. 3 over the training dataset D with CR and p as parameters:

$$\begin{aligned}
\max. \quad & \sum_{(S,O)\in D} [\sum_{i=1}^{n-1} \log \mathrm{CR}(s_i; s_{i+1}|O) + \sum_{j=1}^{n} \log p(s_j|O)] \\
s.t. \quad & \sum_{s,s'} \mathrm{CR}(s; s'|O)p(s|O)p(s'|O) = 1, \forall s, s' \\
& \sum_{s} p(s|O) = 1, \forall s \\
& \mathrm{CR}(s; s'|O) \geq 0, \forall s, s' \\
& p(s|O) \geq 0, \forall s
\end{aligned}$$

First we ignore the last two non-negative inequality constraints. Using Lagrange Multiplier, we obtain the unconstrained objective function:

$$\begin{aligned}
& \sum_{(S,O)\in D} [\sum_{i=1}^{n-1} \log \mathrm{CR}(s_i; s_{i+1}|O) + \sum_{j=1}^{n} \log p(s_j|O)] + \\
& \sum_{s,s'} [\lambda_{s,s'} (\sum_{s,s'} \mathrm{CR}(s; s'|O)p(s|O)p(s'|O) - 1)] \\
& + \sum_{s} [\lambda_s (\sum_{s} p(s|O) - 1)].
\end{aligned}$$

Calculate the first derivative for each parameter and set them to zero, we get the closed form MLE for CR and p:

$$\hat{p}(s|O) = \frac{\#(s,O)}{\sum_s \#(s,O)},$$
$$\hat{CR}(s;s'|O) = \frac{\#(s,s',O)}{\sum_{s,s'} \#(s,s',O)} \Big/ \frac{\#(s,O)}{\sum_s \#(s,O)} \Big/ \frac{\#(s',O)}{\sum_{s'} \#(s',O)}.$$

That is, the MLE of p and CR are just their relative frequencies in the training dataset. Fortunately the non-negative inequality constraints which were ignored in optimization are automatically met.

6.2 Theorems of Co-occurrence Rate

Proof of Partition Operation

Proof.

$$\begin{aligned}
&\mathrm{CR}(X_1;..;X_j)\mathrm{CR}(X_{j+1};..;X_n)\mathrm{CR}(X_1..X_j;X_{j+1}..X_n) \\
&= \frac{p(X_1,..,X_j)}{p(X_1)..p(X_j)} \frac{p(X_{j+1},..,X_n)}{p(X_{j+1})..p(X_n)} \frac{p(X_1,..,X_n)}{p(X_1,..,X_j)p(X_{j+1},..,X_n)} \\
&= \frac{p(X_1,..,X_n)}{p(X_1)..p(X_n)} = \mathrm{CR}(X_1;..;X_n).
\end{aligned}$$

Proof of Reduce Operation

Proof. Since $X \perp\!\!\!\perp Y \mid Z$, we have $p(X,Y|Z) = p(X|Z)p(Y|Z)$, then $p(XYZ) = \frac{p(X,Z)p(Y,Z)}{p(Z)}$. Hence,

$$\mathrm{CR}(X;YZ) = \frac{p(X,Y,Z)}{p(X)p(Y,Z)} = \frac{p(X,Y,Z)}{p(X)p(Y,Z)} = \frac{p(X,Z)}{p(X)p(Z)} = \mathrm{CR}(X;Z).$$

References

1. Altun, Y., Smola, A.J., Hofmann, T.: Exponential families for conditional random fields. In: Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence, UAI '04, pp. 2–9. AUAI Press (2004)
2. Berg-Kirkpatrick, T., Bouchard-Côté, A., DeNero, J., Klein, D.: Painless unsupervised learning with features. In: NAACL, HLT '10, pp. 582–590 (2010)
3. Church, K.W., Hanks, P.: Word association norms, mutual information, and lexicography. Comput. Linguist. **16**(1), 22–29 (1990)
4. Cohn, T.A.: Scaling conditional random fields for natural language processing. Ph.D. thesis, University of Melbourne (2007)
5. Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.: Liblinear: a library for large linear classification. J. Mach. Learn. Res. **9**, 1871–1874 (2008)

6. Ghahramani, Z.: An introduction to hidden Markov models and Bayesian networks. In: Juang, B.H. (ed.) Hidden Markov Models, pp. 9–42. World Scientific Publishing, Adelaide (2002)
7. Hammersley, J.M., Clifford, P.E.: Markov random fields on finite graphs and lattices. Unpublished manuscript (1971)
8. Joachims, T.: Text categorization with support vector machines: learning with many relevant features. In: Proceedings of the 10th European Conference on Machine Learning, ECML '98, pp. 137–142 (1998)
9. Klein, D., Manning, C.D.: Conditional structure versus conditional estimation in NLP models. In: EMNLP '02, pp. 9–16 (2002). http://dx.doi.org/10.3115/1118693.1118695
10. Kudo, T.: CRF++: yet another CRF toolkit, free software, March 2012. http://crfpp.googlecode.com/svn/trunk/doc/index.html
11. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: ICML '01, pp. 282–289 (2001)
12. Le-Hong, P., Phan, X.H., Tran, T.T.: On the effect of the label bias problem in part-of-speech tagging. In: The 10th IEEE RIVF International Conference on Computing and Communication Technologies, pp. 103–108 (2013)
13. McCallum, A., Freitag, D., Pereira, F.C.N.: Maximum entropy markov models for information extraction and segmentation. In: ICML '00, pp. 591–598 (2000)
14. Rabiner, L.R.: A tutorial on hidden markov models and selected applications in speech recognition. In: Proceedings of the IEEE, pp. 257–286 (1989)
15. Smola, A.J., Schölkopf, B.: A tutorial on support vector regression. Stat. Comput. **14**(3), 199–222 (2004)
16. Sutton, C., McCallum, A.: An introduction to conditional random fields. Found. Trends Mach. Learn. **4**(4), 267–373 (2012)
17. Tjong Kim Sang, E.F., De Meulder, F.: Introduction to the CoNLL-2003 shared task: language-independent named entity recognition. In: Proceedings of CoNLL-2003, Edmonton, Canada (2003)
18. Zhu, Z., Hiemstra, D., Apers, P.M.G., Wombacher, A.: Separate training for conditional random fields using co-occurrence rate factorization. Technical report TR-CTIT-12-29, Centre for Telematics and Information Technology, University of Twente, Enschede, October 2012
19. Zhu, Z., Hiemstra, D., Apers, P.M.G., Wombacher, A.: Empirical co-occurrence rate networks for sequence labeling. In: ICMLA 2013, Miami Beach, FL, USA, December 2013, pp. 93–98 (2013)
20. Zhu, Z.: Factorizing probabilistic graphical models using cooccurrence rate, August 2010. arXiv:1008.1566v1
21. Zhu, Z., Hiemstra, D., Apers, P., Wombacher, A.: Comparison of local and global undirected graphical models. In: The 22nd European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, pp. 479–484 (2014)