Constrained Sequence Classification for Lexical Disambiguation

Tran The Truyen, Dinh Q. Phung, and Svetha Venkatesh

Department of Computing, Curtin University of Technology
GPO Box U1987 Perth, Western Australia 6845, Australia
thetruyen.tran@postgrad.curtin.edu.au,
{d.phung,s.venkatesh}@curtin.edu.au

Abstract. This paper addresses lexical ambiguity with focus on a particular problem known as accent prediction, in that given an accentless sequence, we need to restore correct accents. This can be modelled as a sequence classification problem for which variants of Markov chains can be applied. Although the state space is large (about the vocabulary size), it is highly constrained when conditioned on the data observation. We investigate the application of several methods, including Powered Product-of-N-grams, Structured Perceptron and Conditional Random Fields (CRFs). We empirically show in the Vietnamese case that these methods are fairly robust and efficient. The second-order CRFs achieve best results with about 94% term accuracy.

Keywords: constrained sequence classification, lexical disambiguation, Vietnamese accent restoration, conditional random fields.

1 Introduction

Lexical ambiguity is a common problem in natural language processing because lexical analysis is often a first step for high level understanding. In this paper, we focus on a particular problem known as *accent prediction*[1], although the methods can be similarly adapted to other lexical problems such as case prediction and spelling correction (e.g. see [5]).

Accent prediction here refers to the situation where accents are removed (e.g. by some email preprocessing systems), cannot be entered (e.g. by standard English keyboards), or not explicitly represented in the text (e.g. in Arabic). Here we deal with languages that use Roman characters in writing together with additional accent and diacritical marks. Examples are European languages such as Spanish and French (see [8] for comprehensive list) and Asian languages such as Chinese Pinyin and Vietnamese.

The problem often arises because most keyboards today are designed for English, which means without further help, we can only type the Roman alphabets and get an 'approximate' message that is closed to the intended message. The

[1] We use 'accent' to refer to either accents or any diacritical marks.

T.-B. Ho and Z.-H. Zhou (Eds.): PRICAI 2008, LNAI 5351, pp. 430–441, 2008.

practice is popular in email communication, instant messaging and mobile SMS. For example, a Vietnamese sentence: *bạn hãy thăm Việt Nam ngay hôm nay* ('please visit Vietnam today') will be written as an accentless sequence as *ban hay tham Viet Nam ngay hom nay*. Decoding such a message can be quite hard for both human and machine. For instance, the accentless term *ngay* can easily lead to confusion between the original Vietnamese *ngay* ('now' or 'straight') and the plausible alternative *ngày* ('day').

Thus predicting accents is not only useful to recover lost accents, it also reduces typing burden when it provides online suggestions as a shortcut for multiple key combinations. Our approach to this problem is to apply sequence classification techniques. The approach is expected to be more robust than local methods that look only for local context of surrounding words. In addition, training data is not a problem as it is often readily available without any cost of manual labeling. In this paper, we investigate the application of Powered Product-of-N-grams (PPoNs), Structured Perceptron [2] and Conditional Random Fields [7] (CRFs).

The rest of the paper is organised as follows. Related work and background are reviewed in Section 2. The statistical modelling and the PPoNs are proposed in Section 3. Section 4 details the constrained sequence classification methods. In Section 5, we describe the experiments and results for evaluating the proposed methods. Finally, Section 6 concludes the paper.

2 Background

2.1 Previous Work

The most popular approach to lexical ambiguity is corpus-based, where rules and statistical decisions are estimated from the training data. In the case of accent prediction, this is particularly suitable because training data is often readily available without any manual annotation.

A wide range of classification techniques have been used for the accent prediction problem. A comparative study of local methods including most frequent pattern, Bayesian is reported in [14]. These are limited to Spanish and French, where the ambiguity is not very high. For example, using just most frequent accent pattern gives 98.7% accuracy for Spanish. The same approach for Vietnamese, however, achieves only 71.83% accuracy.

Other classification methods include Memory-Based Learning [4], Weighted Finite-State Transducers [9], Hidden Markov Models [12].

The view of accent prediction as sequence classification was considered in [15]. In this work, the Maximum Entropy (MaxEnt) method [1] is used. This is a local method that is adapted for sequences by including label history as features. Our proposal, on the contrary, is to use global methods for classifying sequences.

With respect to lexical analysis there are two main levels: the word level and letter level [8,13]. While the former may be more natural with smoother language models, the latter is more useful when training data is limited (e.g. for languages

with little electronic corpora), or when we have to deal with unknown words (e.g. in medical text [16]).

We are only aware of the published result for Vietnamese in [4], where the best result is only 75.5% term accuracy, much lower than our result of 94.3%. Some software packages are also available, such as AMPad[2] and VietPad[3] but we have not been able to experimentally compare with our methods using the same setup.

2.2 Vietnamese Writing System

The Vietnamese writing system utilises a set of Roman alphabets (the characters 'f', 'j', 'w' and 'z' are not used) and a small set of new symbols and a set of five tonal marks. The five tonal marks are associated with vowels to account for voice stress. Each vowel, and therefore each term, either has zero or one tonal mark. In combination of consonants and vowels, there are about 10^4 unique terms (syllables or unigrams). One or more consecutive terms constitute a word, which is the smallest meaningful text unit. A typical word, especially those in formal writing, has two terms. There are about 10^5 words in the dictionary. Note that word boundaries are not predefined by white spaces as in English. For higher level of understanding, we need to do word segmentation [3], and this is an interesting and important problem of its own right.

3 Problem Modelling

An input sentence $s = (s_1, s_2, ..., s_T)$ can be considered as a distorted version of the original (but unknown) sentence $v = (v_1, v_2, ..., v_T)$, where there is typically a correspondence between the original term v_t and the input term s_t. An exception is that the terms are mistakenly swapped during keying, but this is out of scope of this paper.

In particular, in accent prediction an accentless term is generated by de-accenting the accented counterpart

$$s_t = R(v_t) \ , \tag{1}$$

where $R(v_t)$ is a deterministic function, in that each v_t would yield a unique s_t. In other lexical problems, however, each v_t may correspond to multiple possible assignments of s_t.

The prediction is defined as finding the correct sequence $\hat{v}$ given the distorted sequence s

$$\hat{v}|s = \arg\max_{v \in \mathcal{V}(s)} P(v|s) \tag{2}$$

$$= \arg\max_{v \in \mathcal{V}(s)} P(v)P(s|v) \ , \tag{3}$$

where $\mathcal{V}(s)$ is the space of all possible correct sentences whose distorted form is s.

[2] http://www.echip.com.vn/echiproot/weblh/qcbg/duynghi/ampad/readme.htm
[3] http://vietunicode.sourceforge.net/download/vietpad/

3.1 Powered Product-of-*N*-Grams

In the case of accent prediction we have $P(s|v) = 1$ since the de-accenting is deterministic, and thus

$$\hat{v}|s = \arg \max_{v \in \mathcal{V}(s)} P(v) \ . \tag{4}$$

The problem is now to estimate the language model $P(v)$. In this subsection, we propose to use n-grams as they are still the simplest and effective method. In general, as n increases, we have better language model but we may need a huge data set to have reliable estimate. One effective strategy is to combine n-gram models as follows[4]

$$P(v) = \frac{1}{Z} \prod_n P_n(v)^{w_n} \ , \tag{5}$$

where $Z = \sum_v \prod_n P_n(v)^{w_n}$, $w_n \geq 0$ and $P_n(v)$ are distribution of given by n-gram model. Let us call this method the Powered Product-of-N-grams (PPoNs). The beauty of PPoNs is that the computational complexity is the same as its components, whilst we can adjust the contribution of the components by tuning the extra parameters w_n. The distribution by the PPoNs is often more peaked than the component parts. For example, if the component models agree on a particular sentence v, the PPoNs would yield a very high or very low probability.

The main drawback of the PPoNs model is that we cannot evaluate the normalisation term Z. It prevents estimating the parameters w_n using standard methods such as maximum likelihood. In this work, we manually tune w_n through trials and errors.

4 Constrained Sequence Classification

In standard sequence classification such as part-of-speech tagging we deal with the full state set, and thus with all possible state paths from the start to the end of the sequence. However, in word prediction the state set is particularly large (e.g. in order of $10^4 - 10^5$), it is not practical to perform dynamic programming because the time complexity is quadratic in the size of the state set. Fortunately, in lexical analysis, for each input term, there is a quite small number of corresponding alternatives. We call the set of alternatives by *proposal set*, which can be estimated from a large enough corpus. For example, in accent prediction, the size of proposal sets are less than 25 in the case of Vietnamese, and 5 in Chinese Pinyin. Thus any state path that does not go through those in the proposal set will be eliminated. In other words, the state space is constrained.

[4] One reviewer pointed out that PPoNs are similar to smoothing techniques which approximate n-gram distribution by lower-order distributions. This is interesting because we are originally motivated by the the ensemble methods from the machine learning view.

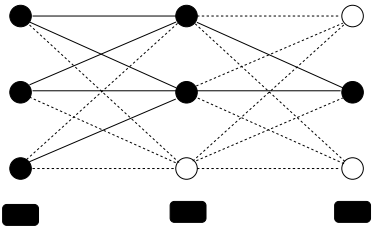

Fig. 1. State paths in constrained first-order Markov chain. Filled circles denotes admissible states, lines denote possible paths, and rounded rectangles denote input terms.

The constraint suggests a better way to model the problem: we do not need to deal with the full state space, rather, for each input sequence, we limit ourselves to the constrained space, *conditioned* on the input sequence. In other words, we estimate the conditional distribution $P(v|s)$ directly.

Denote by $\mathcal{V}(s_t)$ the proposal set for the input term s_t, thus $\mathcal{V}(s) = \mathcal{V}(s_1) \times \mathcal{V}(s_1) \times ... \times \mathcal{V}(s_T)$.

4.1 Conditional Random Fields Modelling

Conditional random fields (CRFs) are particularly suitable for modeling $P(v|s)$. Assuming the $(n+1)$th order Markov chain, the CRF distribution is given as

$$P(v|s) = \frac{1}{Z(s)} \exp(\sum_c \sum_k \lambda_k f_k(v_c, s)) \ , \tag{6}$$

where v_c is the $(n+1)$-gram occurring in the sentence v, $f_k(v_c, s)$ are feature functions, and $Z(s) = \sum_{v \in \mathcal{V}(s)} \exp(\sum_c \sum_k \lambda_k f_k(v_c, s))$.

For example, we can convert the PPoNs into the conditional form by setting

$$f_k(v_{t-n:t}, s) = \log P_n(v_t | v_{t-n}, ..., v_{t-1}) \ . \tag{7}$$

In our study of accent prediction, we do not use the accentless input s in the feature function. The n-grams are used features instead

$$f_k(v_{1:n}, s) = \delta(C(v_{1:n}) > \tau) \ , \tag{8}$$

where $\tau \geq 0$ is the threshold for the number of occurrences $C(.)$, and $\delta(.)$ is the indicator function. The thresholding is important to reduce overfitting and to reduce the number of features significantly because the majority of n-grams appear only once in the corpus.

4.2 Learning CRF Models

Given D training sentences, we learn the parameters of CRF models by maximising the regularised likelihood

$$\hat{\lambda} = \arg\max_{\lambda} \mathcal{L}(\lambda) \ , \tag{9}$$

where

$$\mathcal{L}(\lambda) = \frac{1}{D}\sum_{i=1}^{D} \log P(v^{(i)}|s) - \frac{\|\lambda\|^2}{2\sigma^2} \ . \tag{10}$$

In this study we are interested in online-learning in which parameters are updated after each sentence. This is important in interactive applications where the system may output several possible alternatives and let the user select the one which is most appropriate to his context. Letting the user correct the prediction will allow the system to gradually adapt the model to the domain.

In this study we investigate two online-learning strategies. The first one uses the stochastic gradient ascent [11] to update the parameter as soon as it see the ith sentence. We call this strategy by online maximum likelihood (ML). To smooth the update with fast learning rates and to control the overfitting at the same time, we add a Gaussian regularisation term with mean 0 and variance σ to the likelihood:

$$\bar{\mathcal{L}}(\lambda) = \mathcal{L}(\lambda) - \frac{\|\lambda\|^2}{2\sigma^2} \ . \tag{11}$$

This term basically prevents the weight from being too large, and thus it reduces the tendency of the model to fit the training data too well. It also encourages small parameter changes after seeing each sentence, which is crucial for the stability of the algorithm. The parameter update thus becomes

$$\lambda \leftarrow \lambda + \alpha_i\{\nabla\mathcal{L}(\lambda)\} - \frac{\lambda}{\sigma^2} \ . \tag{12}$$

where $\alpha_i > 0$ is the learning rate. We set $\sigma = 10$ and $\alpha_i = 0.1$ through empirical trials.

The second strategy is based on the Structured Perceptron [2] and is given as

$$\lambda_k \leftarrow \lambda_k + \{f_k(v^i) - f_k(\hat{v}^i)\} \ , \tag{13}$$

where $\hat{v}$ is from Eq. 2. Note that the Structured Perceptron method does not estimate the maximum likelihood but minimises the classification errors. In our implementation, at each step, a sentence is randomly selected from the corpus. We then compute the final parameter by averaging over the parameters learnt after each pass through all data points.

The details of computing the log-likelihood, the optimal $\hat{v}$, and the gradients required for CRF learning are presented in Appendix for clarity.

5 Evaluations

5.1 Corpus and Processing

We collect data from Vietnamese online news sources, split it into a training set of 426K sentences and a test set of 28K sentences. The corpus contains a wide

range of subjects worth reporting[5]. The writing styles vary since the materials come from a dozen news sources.

In order to effectively deal with foreign words, acronyms and non-alphabets, we consider only accentless terms which may correspond to some Vietnamese terms. To obtain the accentless vocabulary, we de-accent the terms in a Vietnamese dictionary. The accentless vocabulary has 1.4K accentless terms, which is much smaller than the typical Vietnamese set of terms (around 10K)[6]. Through de-accenting we obtain the proposal sets, each of which is a set of Vietnamese terms corresponding to a particular accentless term. The number of Vietnamese terms which share the same accentless form ranges from 1 to 24, and is about 4 on average.

For testing, we first perform de-accenting to obtain the accentless form and then decode back the accented form. The decoded text is compared against the original. Here we do not distinguish between upper-case and lower-case. The learning and comparison are done in lower-case.

The performance is measured within the accentless vocabulary. The term accuracy is the portion of restored terms that are correct. A restored sentence is considered correct if all of its restored terms (within the accentless vocabulary) are correct.

From the training data we estimate the unigram, bigram and trigram distributions. There are 7K unique unigrams whose accentless form is in the accentless dictionary. We count a bigram if it occurs and one of the component unigrams is in the unigram list. We obtain a bigram list of size 842K. If we remove those bigrams that happen only once in the corpus, the list is reduced to 465K. Similarly, we count a trigram if it occurs and one of the component unigrams is in the unigram list. This gives 3137K unique trigrams. Removing the trigrams with a single occurrence we obtain a trigram list of size 1264K. We then apply Laplace smoothing, where vocabulary sizes for unigrams, bigrams and trigrams are estimated to be $10^4, 7 \times 10^8$, and 7×10^{13}, respectively. The first estimate is from the 7K unigrams we obtain from the corpus. To estimate the second, recall that the bigrams have two components, one of them must be Vietnamese, and one can be non-Vietnamese. We estimate 10^5 unique non-Vietnamese unigrams in the components of the 842K bigrams from the corpus. Multiplying with the 7K Vietnamese unigrams we obtain 7×10^8. Similarly, multiplying this with 10^5 to obtain the estimate for the trigram vocabulary size.

5.2 Results

For the Powered Product-of-N-grams method described in Section 3, we do not optimise the feature weights $\lambda_1, \lambda_2, \lambda_3$ corresponding to three component models of unigram, bigram and trigram, respectively. Rather, we set the weight manually, and obtain good performance with:

[5] They are: politics, social issues, IT, family & life-style, education, science, economics, legal issues, health, world, sports, arts & culture, and personal opinions.

[6] Note that the evaluation is done *after* restoration, that is, we still use all Vietnamese terms for evaluation.

- first-order PPoNs (unigram & bigram): $w_1 = 1; w_2 = 1$,
- second-order PPoNs (unigram & bigram & trigram): $w_1 = 2; w_2 = 2; w_3 = 1$

First we perform a set of experiments with first-order models, which include the bigrams, the first-order PPoNs and the first-order CRF. One problem with the bigram model is how it handles the unseen bigrams. As the majority of the Vietnamese words used in writing are bigrams, this means that a bigram is not simply random combination of two unigrams. Therefore, most random combinations of two unigrams should have extremely low probability, or at least their probabilities are not equal. The popular Laplace smoothing, on the other hand, tries to assign every unseen bigram an equally small probability under the assumption of prior uniform distribution. This is unrealistic in Vietnamese. To deal with this we assign unseen bigrams with a very low probability, which is practically zero, so that any sequence with unseen bigrams is severely penalised. Although this is not optimal since some plausible bigrams are cut off, it seems to solve the problem. Luckily, the PPoNs does not have this problem, probably because the unseen bigrams will be compensated by the component unigrams.

In the first-order CRF model, we use only the bigram features. For training, we run the Structured Perceptron for 20 iterations and the online ML for 15 iterations over the whole training data set. This is obviously much slower than the bigram and first-order PPoNs models since we need to estimate the bigram distribution using only one run through the data. However, such a cost can be well justified by the higher performance of the CRF model compared with the bigram and the PPoNs as shown in Table 1. In this study, we use 465K bigram features for all the first-order models. The simple unigram model works poorly as expected, and its performance is unacceptable for practical use. The PPoNs, which is just a product of the unigram and the bigram models, works surprisingly well with significant improvement over the bigram model. The CRF model is the winner despite the fact that it uses no more information than that for the PPoNs.

The second set of experiments is performed on second-order models. For the moment, only the second-order PPoNs is used with 7K unigram features, 465K bigram features, and 1264K trigram features. We run the Structured Perceptron

Table 1. Term and sentence accuracy (%) of first/second-order models compared with the baseline unigram model

Model	Term accuracy	Sentence accuracy
Baseline	71.8	6.3
Bigram	90.7	30.9
1st-order PPoNs	92.4	37.6
1st-order CRF (Structured Perceptron)	93.2	38.4
1st-order CRF (Online ML)	93.7	42.0
2st-order PPoNs	93.5	42.7
2st-order CRF (Structured Perceptron)	93.5	41.8
2st-order CRF (Online ML)	94.3	44.8

for 10 iterations and the online ML for 5 iterations. The last rows in Table 1 show the accuracy of the PPoNs and the CRF. The trigram model is not used since it performs fairly poorly, possibly due to the limited corpus. Interestingly, the PPoNs can compensate the poor estimate of the trigrams by using the unigram and bigram components.

Overall for both experiment sets, the CRF trained by online ML performs best. We observe that the Structured Perceptron minimises the error over *training* data quickly since it is specifically designed for this task. The PPoNs, although not as good as the CRF, provides a simple and fast method for model estimation.

A online prototype is available for evaluation at [http://vietlabs.com].

6 Discussion

This paper evaluates a set of constrained sequence classification methods with a particular application in accent prediction. The idea of constrained inference in the context of sequence classification has been proposed earlier [6] in which in the prediction phase some of the labels in the sequence (e.g. accents in our case) are deterministically known. Our work can be considered as an extension to this because we consider a subset of labels as constraints, and use the constraints in both learning and prediction phases. We conjecture that the constraints used in learning can produce better a *conditional* language model for the given restoration task.

Although experimental results on Vietnamese so far indicate that the approach is suitable and can achieve high quality in online news domains, there are open rooms for further improvement. First, there are different genres and writing styles, and it is likely that a sequence of accentless terms can correspond to several plausible Vietnamese sequences, depending on the context of use. A very challenging domain is creative writing, especially in poetry, where the authors make deliberate use of word reordering and repetition to achieve stylistic and artistic effect. The most challenging form is perhaps spoken language, especially in the online environments such as chatting and SMS, where the use of language is largely distorted due to the constraints of writing space and personal interests.

An issue not addressed in this work is the analysis of syntax and semantics. It is likely that the analysis will provide more consistent and grammatical results as well as coherence within and between sentences in the document. Through the CRF framework, for example, it is possible to incorporate a richer set of features to address the correlation between sentences in the same paragraph. Also, we can create different models to address different linguistic aspects and then combine them together in the PPoNs approach.

References

1. Berger, A.L., Della Pietra, S.A., Della Pietra, V.J.: A Maximum Entropy approach to natural language processing. Computational Linguistics 22, 39–71 (1996)
2. Collins, M.: Discriminative training methods for hidden Markov models: Theory and experiments with the perceptron algorithm. In: Conference on Empirical Methods in Natural Language Processing (EMNLP) (2002)

3. Dien, D., Kiem, H., Toan, N.V.: Vietnamese Word Segmentation. In: NLPRS 2001, pp. 749–756 (2001)
4. De Pauw, G., Wagacha, P.W., de Schryver, G.M.: Automatic Diacritic Restoration for Resource-Scarce Languages. In: Matoušek, V., Mautner, P. (eds.) TSD 2007. LNCS (LNAI), vol. 4629. Springer, Heidelberg (2007)
5. Golding, A.R., Roth, D.: Winnow-Based Approach to Context-Sensitive Spelling Correction. Machine Learning 34, 107–130 (1999)
6. Kristjannson, T., Culotta, A., Viola, P., McCallum, A.: Interactive information extraction with constrained conditional random fields. In: 19th National Conference on Artificial Intelligence (AAAI), pp. 412–418 (2004)
7. Lafferty, J., McCallum, A., Pereira, F.: Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In: International Conference on Machine learning (ICML), pp. 282–289 (2001)
8. Mihalcea, R., Nastase, V.: Letter level learning for language independent diacritics restoration. In: International Conference On Computational Linguistics, pp. 1–7 (2002)
9. Nelken, R., Shieber, S.M.: Arabic Diacritization Using Weighted Finite-State Transducers. In: ACL Workshop on Computational Approaches to Semitic Languages (2005)
10. Rabiner, L.R.: A tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of the IEEE 77, 257–286 (1989)
11. Robbins, H., Monro, S.: A stochastic approximation method. The Annals of Mathematical Statistics 22, 400–407 (1951)
12. Simard, M., Deslauriers, A.: Real-time automatic insertion of accents in French text. Natural Language Engineering 7, 143–165 (2001)
13. Wagacha, P., De Pauw, G., Githinji, P.: A grapheme-based approach for accent prediction in Gıkuyu. In: 5th International Conference on Language Resources and Evaluation, Genoa, Italy, pp. 1937–1940 (2006)
14. Yarowsky, D.: A comparison of corpus-based techniques for restoring accents in Spanish and French text. In: Natural Language Processing Using Very Large Corpora. Springer, Heidelberg (1999)
15. Zitouni, I., Sorensen, J.S., Sarikaya, R.: Maximum Entropy based restoration of Arabic diacritics. In: 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL, pp. 577–584 (2006)
16. Zweigenbaum, P., Grabar, N.: Restoring accents in unknown biomedical words: application to the French MeSH thesaurus. International Journal of Medical Informatics 67, 113–126 (2002)

A Constrained Inference

In this appendix, we provide a general account for inference in first and second-order models with constrained state spaces.

Inference in the n-gram models is mostly Viterbi decoding (as in Eq. 2) since maximum likelihood learning of such models is done through frequency counting. This also applies for the PPoNs method since we do not perform further learning after estimating the n-gram components. However, inference in the CRF is needed for estimating the partition function (as in Eq. 6) and the feature expectation as shown in Section 4.2.

A.1 First-Order Models

Associated with each node t in the first-order Markov chain is a positive potential $\phi(v_t, s)$ to account for the statistics of the distorted term v_t given the input s. Similarly, associated with each edge is a positive potential $\psi(v_{t-1}, v_t, s)$ to account for the statistics of the transition from v_{t-1} to v_t. The correspondence between these potentials and the proposed models are as follows

- In the unigram model, $\phi(v_t, s) = P(v_t)$,
- In the bigram model, $\phi(v_1, s) = P(v_1)$, $\phi(v_t) = 1$ for $t > 1$, and $\psi(v_{t-1}, v_t, s) = P(v_t|v_{t-1})$,
- In the bigram PPoNs model, $\phi(v_t, s) = P(v_t)^{\lambda_1}$, $\psi(v_{t-1}, v_t, s) = P(v_t|v_{t-1})^{\lambda_2}$, and
- In the first-order CRF model, $\phi(v_t, s) = \exp(\lambda_k f_k(v_t, s))$, and $\psi(v_{t-1}, v_t, s) = \exp(\lambda_k f_k(v_{t-1}, v_t, s))$.

For the first-order Markov chains, inference can be done using the forward-backward procedure.The forward α_t is defined recursively as

$$\alpha_t(v_t \in \mathcal{V}(s_t)|s) \propto \sum_{v_{t-1} \in \mathcal{V}(s_{t-1})} \alpha_{t-1}(v_{t-1}|s)\phi(v_{t-1}, s)\psi(v_{t-1}, v_t, s) \ , \quad (14)$$

where $\alpha_1(v_1 \in \mathcal{V}(s_1)|s) = 1$; Similarly, the backward β_t is

$$\beta_t(v_t \in \mathcal{V}(s_t)|s) \propto \sum_{v_{t+1} \in \mathcal{V}(s_{t+1})} \beta_{t+1}(v_{t+1}|s)\phi(v_{t+1}, s)\psi(v_t, v_{t+1}, s) \ . \quad (15)$$

For the feature expectations in CRFs (as in Eq. 12), we need to compute the marginal

$$P(v_t|s) \propto \alpha_t(v_t|s)\beta_t(v_t|s)\phi(v_t, s) \ , \quad (16)$$

and the joint marginals

$$P(v_t, v_{t+1}|s) \propto \alpha_t(v_t|s)\beta_t(v_{t+1}|s)\phi(v_t, s)\phi(v_{t+1}, s)\psi(v_t, v_{t+1}, s) \ . \quad (17)$$

The complexity of these procedures is therefore $\mathcal{O}(T|S|^2)$ where $|S| = \max_t |\mathcal{V}(s_t)|$.

To do restoration we can use the Viterbi decoding [10], paying attention to the constraints. Alternatively, we can use the equivalent Pearl's max-product algorithm where the summations in Eqs. 14 and 15 are replaced by the maximisations. Similar to the forward-backward, this max-product algorithm takes $\mathcal{O}(T|S|^2)$ time.

A.2 Second-Order Model

Now we need to take the trigrams into account by using the extension of the edge potential $\psi(v_{t-1}, v_t, s)$ to incorporate the state v_{t-2}. With a slight abuse of notation, denote by $\psi(v_{t-2}, v_{t-1}, v_t, s)$ the trigram potentials.

- In the trigram model, $\psi(v_{t-2}, v_{t-1}, v_t, s) = P(v_t|v_{t-1}, v_{t-2})$,
- In the bigram PPoNs model, $\psi(v_{t-2}, v_{t-1}, v_t, s) = P(v_t|v_{t-1}, v_{t-2})^{\lambda_3}$, and
- In the second-order CRF model, $\psi(v_{t-2}, v_{t-1}, v_t, s) = \exp(\lambda_k f_k(v_{t-2}, v_{t-1}, v_t, s))$.

Efficient inference in the second-order Markov chains is a bit more tricky since we do not have the chains. First, we need to convert the second-order Markov chains into the first-order equivalence. The conversion is done by joining two successive nodes $\{v_{t-1}, v_t\}$ into a composite-node $V_{t-1} = \{v_{t-1}, v_t\}$. Let the composite-node potential be $\phi(V_{t-1}, s) = \phi(v_{t-1})\psi(v_{t-1}, v_t, s)$ and the composite-edge potential be and $\psi(V_{t-1}, V_t, s) = \psi(v_{t-1}, v_t, v_{t+1}, s)$. Given these potentials, it can be seen that we now have a new first-order Markov chain with the combined state space:

$$V_{t-1} \in \mathcal{V}(s_{t-1}, s_t) = \mathcal{V}(s_{t-1}) \times \mathcal{V}(s_t) \ . \tag{18}$$

The naïve implementation of this Markov chain takes $\mathcal{O}((T-1)|S|^4)$ time in this combined state space. However, by paying attention to the fact that the two composite-states $V_{t-1} = \{v_{t-1}, v_t\}$ and $V_t = \{v_t, v_{t+1}\}$ share the same term v_t, we can implement the forward-backward procedure in $\mathcal{O}((T-1)|S|^3)$ time using

$$\alpha_t(V_t|s) = \sum_{x_{t-1} \in \mathcal{V}(x_{t-1})} \alpha_{t-1}(V_{t-1}|s)\phi(V_{t-1}, s)\psi(V_{t-1}, V_t, s) \ ,$$
$$\beta_t(V_t|s) = \sum_{x_{t+2} \in \mathcal{V}(x_{t+2})} \beta_{t+1}(V_{t+1}|s)\phi(V_{t+1}, s)\psi(V_t, V_{t+1}, s) \ ,$$

and the joint marginals are computed as

$$P(V_t|s) \propto \alpha_t(V_t|s)\beta_t(V_t|s)\phi(V_t, s) \ ,$$
$$P(V_t, V_{t+1}|s) \propto \alpha_t(V_t|s)\beta_t(V_{t+1}|s)\phi(V_t, s)\phi(V_{t+1}, s)\psi(V_t, V_{t+1}, s) \ .$$