N-grams for Conditional Random Fields
or a Failure-transition(ϕ) Posterior for Acyclic FSTs

Patrick Lehnen, Stefan Hahn, Hermann Ney

Abstract

Freely available software packages for the training of Conditional Random Fields, e.g. CRF++, do not support longer n-grams than bigram, which can be attributed to the fact that training CRFs in original notation has a polynomial computation time dependence on the target vocabulary size and an exponential dependence on the n-gram size. We transfer the back-off idea from language models to CRFs. We realized the software with Finite State Transducers, where we modified the calculation of the posterior algorithm. To implement the back-off scheme, we applied Failure-transitions(ϕ) known from OpenFST. Proof of concept is given on the semantic tagging task MEDIA and on the grapheme-to-phoneme (G2P) conversion tasks NETalk and Celex, showing that computational time increases much below the size of the target vocabulary and showing error rate reduction on the G2P tasks.

Index Terms: CRF, n-gram, trigram, FST, posterior, NLU, G2P

1. Introduction

Conditional Random Fields (CRFs) [1] have become a very popular and powerful approach to natural language processing tasks over the last decade. They provided state of the art results, e.g. in concept tagging [2], name transliteration [3], and grapheme-to-phoneme conversation (G2P) [4], but are not applicable to tasks with target vocabularies in the range of 1,000-100,000 words or longer n-gram contexts than bigrams, which is due to their polynomial computation time dependence on the size of the target vocabulary $|V_T|$ and exponential dependence on the n-gram length $\delta$. With bigrams, the computational time dependence is quadratic in the size of the target vocabulary. The algorithm proposed in this paper weakens this restriction. The computational complexity is now dominated by the average number of n-grams per (n-1)-gram.

Proof of concept is given on the semantic tagging corpus MEDIA [5] and the G2P corpora NETalk [6] and Celex [7]. Both tasks are machine learning tasks from source sequence $s^N_1$ to target sequence $t^N_N$, where monotone alignments are sufficient. An example from the MEDIA corpus is the sentence “j' veux une chambre double” (“I would like to have a double room”), which is tagged with "@null{ Je veux } @number_rooms { une } @room_type { chambre double }". The corpus is shipped with a given alignment. So we can avoid the alignment problem by adopting the so-called BIO scheme [8] (“begin” (B) and “inside” (I) markers, cf. [2]). G2P is translation of letter sequences (words) to phoneme sequences. The NETalk corpus is shipped with a given alignment, too, while it has to be generated for the Celex corpus.

In training CRF models for these corpora one can restrict the CRF features to features only seen in the training corpus, resulting in very sparse models having per source symbol only some features per target symbol. Especially in the context of n-gram features the number of n-grams per (n-1)-gram is very low. In traditional language modelling (LM) it is common to “back-off” all not seen transitions to smaller contexts. Using the Failure-transitions $\phi$ for Finite State Transducers (FSTs) presented in [9], this “backing-off” can be used in the context of CRFs. In [4], a description of reducing the computation time by taking into account the bigram sparsity is given, but they do not extend the idea to longer n-grams. [10] claim to train discriminative LMs with arbitrary n-gram length using CRFs and the perceptron algorithm. They use mainly the algorithm ExpCount from the GRM library [9], where we do not find a variant able to use the failure transitions. They trained the CRFs on ASR lattices. We assume that ExpCount was applied after the LM was applied to that lattice with a composition algorithm which can take failure transitions into account (e.g. implemented in OpenFST [11]). We therefore derived a variant of that ExpCount algorithm for failure transitions, called in this publication posterior algorithm.

In the next section we define the notation for CRFs, followed by a motivation of the sparse forward/backward algorithm for bigrams in Sec. 3, and the actual introduction to our algorithm in Sec. 4. An overview of used features is given in Sec. 5. Finally, we present the experimental results in Sec. 6 and conclude in Sec. 7.

2. Conditional Random Fields

Linear Chain Conditional Random Fields (CRFs) introduced in [1] are defined as the conditional probability of a target sequence $t^N_1 = t_1,\ldots,t_N$ given a source sequence $s^N_1 = s_1,\ldots,s_N$ using a log-linear representation:

$$p(t^N_1 | s^N_1) = \frac{\prod_{n=1}^{N} e^{H(t^N_{n-\delta},s^N_1)}}{\sum_{t^N_1} \prod_{n=1}^{N} e^{H(t^N_{n-\delta},s^N_1)}} \quad (1)$$

$$H(t^N_{n-\delta},s^N_1) = \sum_{l=1}^{L} \lambda_l h_l(t^N_{n-\delta-1},t_n,s^N_1) \quad (2)$$

$H(t^N_{n-\delta},s^N_1)$ defines position dependent feature functions $h_l(t^N_{n-\delta-1},t_n,s^N_1)$ (details in Sec. 5). The training criteria over a training dataset $\{ (t^N_1)_k, (s^N_1)_k \}_{k=1}^{K}$ is given by the maximization of the conditional log-likelihood $L$

$$L = \sum_{k=1}^{K} \log p(t^N_1)_k | (s^N_1)_k) - c_1 ||\lambda^M||_1 - c_2 ||\lambda^M||_2^2 \quad (3)$$
using L1- and L2-regularization constants $c_1$, $c_2$, while the decision criterion is given by the maximization of the sentence-wise probability $p(s^n | t^n)$. Eq. 3 can be written using forward $\alpha$ and backward $\beta$ weights. For the sake of clarity the regularizations and the dependence on $s^n$ are omitted. For details about the derivation of the regularization cf. e.g. [12].

$$\frac{\partial L}{\partial \lambda_n} = \sum_{k=1}^{K} \sum_{m=1}^{N} h(t_{n-\delta}^{m}, t_n, s_n^{N})$$

$$+ \sum_{t_{n-1}, t_n} p(t_{n-1}, t_n) h_1(t_{n-\delta}^{m}, t_n, s_n^{N})$$

with

$$p(t_{n-1}, t_n) = \frac{1}{Z} \alpha_n(t_{n-1}) e^{H(t_{n-\delta}^{m}, s_n^{N})} \beta_{n+1}(t_n)$$

$$\alpha_n(t_{n-1}) = \sum_{t_{n-2}} e^{H(t_{n-\delta}^{m-1}, s_n^{N})} \alpha_{n-1}(t_{n-2})$$

$$\beta_{n+1}(t_n) = \sum_{t_{n+1}} e^{H(t_{n-\delta}^{m+1}, s_n^{N})} \beta_{n+2}(t_{n+1})$$

$$\beta_{N+1} = 1, \quad \alpha_1 = 1, \quad Z = \beta_0$$

3. Modified Forward/Backward

If the potential function $H$ can be reduced to a smaller context $H(t_{n-\delta}^{m}, s_n^{N}) = H(t_{n-\delta}^{m}, s_n^{N})$, e.g. if the $\delta$-gram features are zero, it is possible to reduce the number of computations. We will call the set were the equality is given $t_{n-\delta} \in S$. In [4] it is described how to reduce the computational cost for bigrams. The trick can be rephrased as

$$\alpha_n(t_{n-1}) = \sum_{t_{n-2} \not\in S} e^{H(t_{n-\delta}^{m-1}, s_n^{N})} \alpha_{n-1}(t_{n-2})$$

$$+ \sum_{t_{n-2} \in S} e^{H(t_{n-\delta}^{m-1}, s_n^{N})} \alpha_{n-1}(t_{n-2})$$

$$= \sum_{t_{n-2} \not\in S} \left( e^{H(t_{n-\delta}^{m-1}, s_n^{N})} \alpha_{n-1}(t_{n-2}) - e^{H(t_{n-\delta}^{m-1}, s_n^{N})} \right) \alpha_{n-1}(t_{n-2})$$

$$+ \sum_{t_{n-2} \in S} e^{H(t_{n-\delta}^{m-1}, s_n^{N})} \alpha_{n-1}(t_{n-2})$$

where the second sum can be precomputed. This idea can be extended to arbitrary $\delta$s as described in the next section.

4. A Failure-transition ($\phi$) Posterior

To correctly model statistical language models (SLMs) in FSTs, one needs an additional special variant of the $\epsilon$-label. This label is usually called failure or $\phi$ (cf. [11, 9]). The failure transition excludes succeeding transitions, which are already given in the state omitting the failure transition. A SLM is realized as in Fig. 1. E.g. state 1 could have history (B,C) and omitting the trigram A, for the label B only a bigram and for the label C only a unigram is known.

We have modified the posterior calculation with respect to the Failure-transitions inspired by the idea from [4]. The resulting algorithm is presented in Figs. 2, 3, and 4. Without Failure-transitions it is equivalent to the standard posterior algorithm as the $\epsilon$-arc procedures are not used then. Forward/Backward projections keep track of the Failure-transitions restriction by the set $E$, by recursively calling the procedure state with $E$ including all already seen input/output-label combinations. With a semiring supporting an operation $\oplus$ with $a \oplus b = c \iff a = b \ominus c$ (e.g. the log-semiring) the computational cost can be significantly reduced, since the recursion of state is restricted to one level. To calculate posterior weights at the failure-transitions and at the transitions succeeding a failure-transition correctly, it is now necessary to keep track of one additional back-off weight $\beta'$ per weight and additional forward weights $\alpha'$ per transition. Using efficient structures, the $\alpha$'s have to be kept in memory only if they are really needed. The depth first search is used to calculate only those states, where the potential of successor/predecessor states is already finished. The given algorithm currently does not take paths with cycles into account, since these cycles are not needed for our task.

5. Features

In the described experiments, the feature functions $h_1(t_{n-\delta}^{m-1}, t_n, s_n^{N})$ are binary features ($\epsilon \in \{0,1\}$), having emission features depending on source words and the current target word $(t_n, s_m)$ with some $m$, “and”-combinations of these features $(t_n, s_m)$, subword units (first/last x letters), and transition features $(t_{n-\delta}^{m-1}, t_n)$ of different length $\delta$. To gain sparseness we only selected features seen at least one time in the training set. In CRFs usually all atomic features are added (bigram and trigram at the same time), but in traditional SLMs a bigram is only used if the corresponding trigram is not used (“backing-off”). To investigate the impact of this decision we conducted experiments with this constraint.

6. Experimental Results

To verify the computational complexity of the presented algorithm experimentally, two types of tasks have been chosen, namely concept tagging and grapheme-to-phoneme (G2P) conversion. For concept tagging, we use the well-known French MEDIA corpus, which has been used in a number of state-of-the-art evaluations [5]. It consists of Wizard-of-Oz dialogues in the domain of negotiation of tourist services. For G2P, we conducted experiments on the English NETtalk [6] and Celex [7] corpora. Both tasks have a relatively small output vocabulary between 50 and 100 words and are thus well suited for this kind of experiment, where a blow-up in computational complexity is assumed w.r.t. this size. Also, the two G2P corpora have been used in state-of-the-art research papers, e.g. [2, 13, 14]. Since the NETtalk corpus does not provide a development set, a part
of the training set has been set aside for tuning. We apply the same splitting as in the aforementioned publications. The most relevant corpus statistics are given in Table 1.

Since performance is an important aspect of the algorithm, we also quantify the time for the training of the CRF models. All our computations have been run on a SUN Grid Engine cluster of computers with AMD Opteron CPUs with 2.2GHz to 2.8GHz and were parallelized on corpus level (utterances selected by modulo) on 10 CPUs.

The results on the MEDIA corpus are given in Table 2. The baseline setup is given in the first line of the table. This is a highly tuned baseline using additionally to the standard bigram and lexical features also a capitalization feature as well as subword-features on source side (see [2] for more details). 50 training iterations are used to obtain convergence and the total training time is roughly one hour per iteration. If we extend the context length on target side to 3grams, the computational time increases by a factor of seven, which is clearly not an exponential increase. In the latter experiments, for each seen trigram feature, the bigram features have always been considered as well. In the experiments in parentheses, we constrain the features like in language modeling (cf. Sec. 5). Unfortunately, a longer context does not seem to help for the concept tagging task. To proof the feasibility of the algorithm, we extended the n-gram length to four, which results in a factor of 23 w.r.t computational time per training iteration. As expected, the increased context did not result in a performance gain.

For the G2P task, besides the conceptual proof that it is feasible to use larger context lengths on target side, there is a gain in performance. As presented in Tab. 2 for the NETtalk corpus, trigram and 4-gram features lead to an improvement in WER for trigrams and 4-grams. Again the features constraint as well. In the experiments in parentheses, we constrain the features like in language modeling (cf. Sec. 5). Unfortunately, a longer context does not seem to help for the concept tagging task. To proof the feasibility of the algorithm, we extended the n-gram length to four, which results in a factor of 23 w.r.t computational time per training iteration. As expected, the increased context did not result in a performance gain.

Figure 3: Single Source Shortest Distance Algorithm in forward orientation. The potentials α, and transition wise potentials α′ are updated in the inverted order a depth first search finalizes the states of G, df(G). For every state the method state is called using the methods arc and failArc, which calls state recursively. The set E keeps track of already found transitions. The faster methods arc′ and failArc′ can be used if the used semiring supports an operation ⊕ with a ⊕ b = c ⇔ a = b ⊕ c.

Table 1: Elementary corpus statistics for the MEDIA corpus, the NETtalk corpus, and the Celex corpus. To keep presentation clear we only include the key numbers. Details can be found in the original publications.

<table>
<thead>
<tr>
<th></th>
<th>MEDIA</th>
<th>NETtalk</th>
<th>Celex</th>
</tr>
</thead>
<tbody>
<tr>
<td># training utt</td>
<td>12,908</td>
<td>13,935</td>
<td>39,995</td>
</tr>
<tr>
<td># dev. utt</td>
<td>1,259</td>
<td>1,071</td>
<td>5,000</td>
</tr>
<tr>
<td># test utt</td>
<td>3,005</td>
<td>5,002</td>
<td>15,000</td>
</tr>
<tr>
<td>source vocabulary</td>
<td>2,210</td>
<td>26</td>
<td>26</td>
</tr>
<tr>
<td>target vocabulary</td>
<td>99</td>
<td>50</td>
<td>53</td>
</tr>
</tbody>
</table>
Table 2: Results for different n-gram lengths on Media, NETtalk 15k, and Celex. Besides the number of active features and the training time per iteration, the concept error rate (Media) / phoneme error rate (G2P) as well as the sentence error rate (Media) / word error rate (G2P) is presented. Numbers in parentheses refer to constrained features like in language modeling ("backing-off", cf. Sec. 5).

<table>
<thead>
<tr>
<th>n</th>
<th>#features</th>
<th>time per iter. [h]</th>
<th>CER/PER[%]</th>
<th>SER/WER[%]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Dev</td>
<td>Eva</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Media</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>36,095</td>
<td>1</td>
<td>12.7</td>
<td>11.9</td>
</tr>
<tr>
<td>3</td>
<td>41,005</td>
<td>7</td>
<td>12.6 (13.6)</td>
<td>12.2 (12.7)</td>
</tr>
<tr>
<td>4</td>
<td>50,630</td>
<td>23</td>
<td>15.5 (15.3)</td>
<td>15.0 (14.7)</td>
</tr>
<tr>
<td>NETtalk</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>803,092</td>
<td>1</td>
<td>7.7</td>
<td>8.1</td>
</tr>
<tr>
<td>3</td>
<td>813,069</td>
<td>14</td>
<td>7.6 (8.2)</td>
<td>7.9 (8.4)</td>
</tr>
<tr>
<td>4</td>
<td>836,422</td>
<td>69</td>
<td>7.4 (8.7)</td>
<td>7.8 (8.5)</td>
</tr>
<tr>
<td>Celex</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1,288,832</td>
<td>2</td>
<td>3.0</td>
<td>2.9</td>
</tr>
<tr>
<td>3</td>
<td>1,298,632</td>
<td>27</td>
<td>2.8 (3.0)</td>
<td>2.8 (2.9)</td>
</tr>
<tr>
<td>4</td>
<td>1,330,567</td>
<td>186</td>
<td>2.7</td>
<td>2.6</td>
</tr>
</tbody>
</table>

Figure 4: Final posterior algorithm using forward and backward. The command after “if arc’” may be used instead of the command after “default” if arc’ and failArc’ have been used.

7. Conclusions

In this paper we presented an extension of the posterior algorithm on Finite State Transducers for Failure-Transitions. With the help of these transitions it was possible to train target n-gram features up to the length of 4-grams. The algorithm does not restrict the maximal n-gram size, thus it should be possible to train even 5- or 6-grams. The computation time reduction was verified on three corpora, namely one from a semantic tagging task and two from the the task of grapheme-to-phoneme conversion. On the grapheme-to-phoneme tasks, a gain in performance could be observed additionally.

8. Acknowledgements

This work was partly realized as part of the Quaero Programme, funded by OSEO, French State agency for innovation.

9. References


