POWERFUL EXTENSIONS TO CRFS FOR GRAPHEME TO PHONEME CONVERSION

Stefan Hahn *Patrick Lehnen* *Hermann Ney*

{hahn,lehnen,ney}@cs.rwth-aachen.de
Human Language Technology and Pattern Recognition
Computer Science Department, RWTH Aachen University, 52056 Aachen, Germany

ABSTRACT

Conditional Random Fields (CRFs) have proven to perform well on natural language processing tasks like name transliteration, concept tagging or grapheme-to-phoneme (g2p) conversion. The aim of this paper is to propose some extension to the state-of-the-art CRF systems for these tasks. Since the number of features can grow rapidly, a method for features selection is very helpful to boost performance. A combination of L1 and L2 regularization (elastic net) has been adopted and implemented within the Rprop optimization algorithm. Usually, dependencies on the target side are limited to bigram dependencies since the computational complexity grows exponentially with the history length. We present a modified CRF decoding where a conventional language model on target side is integrated into the CRF search process. Thus, larger contexts can be taken into account. Besides these two main parts, the already published margin-extension to the CRF training criterion has been adopted.

Index Terms— G2P, CRF, LM, Margin, Elastic-Net, L1

1. INTRODUCTION

Conditional Random Fields (CRFs) provide a powerful model for natural language processing tasks like grapheme-to-phoneme conversion [1], name transliteration [2] or concept tagging [3]. This discriminative modelling approach has become more and more popular within the speech community due to its nice theoretic properties as well as state-of-the-art results on a large number of NLP tasks. Nevertheless, there are some drawbacks to this model. CRFs permits to use overlapping features leading in some tasks (e.g. g2p) very fast to huge feature sets with 100M-1G features. Quasi-Newton update methods like L-BFGS need to keep the feature parameters λ_1^M, the gradient of the conditional log-likelihood, and an approximation to the Hessian in memory. It is obvious that there is a need to select a set of really useful features. Taking into account that we only have about 50 output labels and 26 input labels, most features are rarely seen and cannot be trained properly. Thus, one can expect even a gain in performance by selecting only useful features.

In [1], elastic nets are proposed and included in orthant-wise quasi-Newton (OWL-QN) algorithm, stochastic gradient descent, and block coordinate descent. Although this methods are clearly powerful, they are expensive in computational costs, memory consumption and time to develop software to use this algorithm.

Additionally, the computational complexity of CRFs grows exponentially with the context length on target side. Thus, only bigram dependencies are computationally feasible. Within this work, methods to cope with these drawbacks are presented. Instead of the popular L-BFGS, the Rprop [4] optimization algorithm is utilized and a method similar to the elastic net has been implemented. To cope with the restricted context length on target side, we propose an integration of a classical language model on target side within the search process. Additionally, small modifications to CRFs are discussed like the integration of a margin into the training criterion, which gives some performance improvements [5], [3].

To assess the quality of the methods, their performance is evaluated on the NETtalk task.

The next section presents our baseline CRF system, which already includes some tweaks compared to the standard CRF. Sec. 3 presents our new feature-selection method implemented within Rprop. In Sec. 4, the integration of a classical LM into the CRF search is presented. The following section gives experimental results. The paper concludes with Sec. 6.

2. CRF WITH MARGIN-EXTENSION

Linear Chain Conditional Random Fields (CRFs) as introduced in [6] are defined as the conditional probability of a target sequence $t_1^N = t_1, \ldots, t_N$ given a source sequence $s_1^N = s_1, \ldots, s_N$ using a log-linear representation:

$$p(t_1^N|s_1^N) = \frac{\exp H(t_1^N, s_1^N)}{\sum_{\tilde{t}_1^N} \exp H(\tilde{t}_1^N, s_1^N)} \quad (1)$$

$$H(t_1^N, s_1^N) = \left(\sum_{n=1}^{N} \sum_{m=1}^{M} \lambda_m h_m(t_{n-1}, t_n, s_1^N) \right) \quad (2)$$

$H(t_1^N, s_1^N)$ represents the sentence-wise accumulation of position dependent and binary feature functions $h_m(t_{n-1}, t_n, s_1^N)$. The feature functions return "1" iff a given configuration is found in the parallel sequences. In the experiments, three sets of feature functions were used: lexical features $(t_n = t', s_{n+\epsilon} = s')$, a bigram feature $(t_{n-1} = t'', t_n = t')$,

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

Input: Last and current lambdas $\{\lambda_1^M\}_{i-1}$, $\{\lambda_1^M\}_i$, current step sizes $\{s_1^M\}_i$, last and current gradient of the objective function $\{\nabla_{\lambda_1^M} L\}_{i-1}$, $\{\nabla_{\lambda_1^M} L\}_i$
Output: New lambdas $\{\lambda_1^M\}_{i+1}$, new step sizes $\{s_1^M\}_{i+1}$
for $m \in 1, \ldots, M$:
  if $\{\frac{\partial L}{\partial \lambda_m}\}_{i-1} \cdot \{\frac{\partial L}{\partial \lambda_m}\}_i > 0$:
    $s_{m,i+1} = \min(s^+ \cdot s_{m,i}, s_{max})$
    $\lambda_{m,i+1} = \lambda_{m,i} - sign(\{\frac{\partial L}{\partial \lambda_m}\}_i) \cdot s_{m,i+1}$
  else if $\{\frac{\partial L}{\partial \lambda_m}\}_{i-1} \cdot \{\frac{\partial L}{\partial \lambda_m}\}_i < 0$:
    $s_{m,i+1} = \max(s^- \cdot s_{m,i}, s_{min})$
    $\lambda_{m,i+1} = \lambda_{m,i-1}$
    $\{\frac{\partial L}{\partial \lambda_m}\}_i = 0$
  else if $\{\frac{\partial L}{\partial \lambda_m}\}_{i-1} \cdot \{\frac{\partial L}{\partial \lambda_m}\}_i == 0$:
    $s_{m,i+1} = s_{m,i}$
    $\lambda_{m,i+1} = \lambda_{m,i} - sign(\{\frac{\partial L}{\partial \lambda_m}\}_i) \cdot s_{m,i}$

Fig. 1. Rprop Algorithm as proposed in [4]. s^+, s^-, s_{max}, s_{min} are configuration variables typically defined as $s^+ = 1.2$, $s^- = 0.5$, $s_{max} = 50$, $s_{min} = 0$.

and a huge set of and-combinations of the lexical features resulting in m-grams on source side $(t_n = t', s_{n+\epsilon-\delta}^{n+\epsilon+\delta} = s')$.

The training criterion on a training set $\{\{\bar{t}_1^N\}_k, \{s_1^N\}_k\}_{k=1}^K$ is given by maximization of the conditional log-likelihood L with respect to λ_1^M:

$$L = \sum_{k=1}^{K} \log p(\{\bar{t}_1^N\}_k | \{s_1^N\}_k) - r(\lambda_1^M) \tag{3}$$

using a regularization function r described in detail in Sec. 3, while the decision criterion is given by the maximization of the sentence wise probability $p(t_1^N | s_1^N)$.

Recently, an extension to CRFs called the margin-based extension has been introduced in [5]. It is based on the idea to integrate the training of SVMs and CRFs, called MMI there and results in a modification of the potential function H to

$$\hat{H}(t_1^N, s_1^N) = H(\tilde{t}_1^N, s_1^N) - \rho \mathcal{A}(\tilde{t}_1^N, \bar{t}_1^N) \tag{4}$$

with the old potential function H from Eq. 2 and a margin score set to the word accuracy $\mathcal{A}(t_1^N, \bar{t}_1^N) = \sum_{n=1}^N \delta(t_n, \bar{t}_n)$ between the hypothesis t_1^N and the reference $\bar{t}_1^N$, scaled by $\rho \geq 0$. In our experiments ρ was set to 1 and the weight of both summands of $\hat{H}$ was tuned by the regularization function r. The modification is only included in the training of parameters.

3. ELASTIC NET FOR RPROP

We like to propose an extension to the very simple Rprop-Algorithm. Rprop uses only the sign of the current and last gradient of the objective function L (Eq. 3). An abstract of the algorithm described in [4] is given in Fig. 1. Elastic-Net is a combination of L2 and L1 regularization $r(\lambda_1^M) = c_2||\lambda_1^M||_2^2 + c_1||\lambda_1^M||_1$ with the 2- and 1-Norm $||\cdot||_{1/2}$. Due to the convexity property of CRFs which implies $\partial^2 L/\partial \lambda_m^2 < 0$, the gradient of L can be approximated at zero as

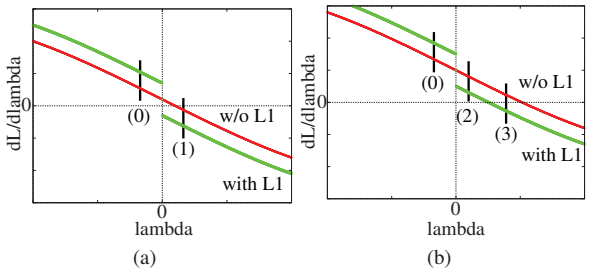

Fig. 2. Sketches of the gradient of the objective function from Eq. 3 with equal L1 regularization c_1 but different offset.

$$\frac{\partial L}{\partial \lambda_m} \approx \left.\frac{\partial L}{\partial \lambda_m}\right|_{\lambda_m=0} - \left.\frac{\partial^2 L}{\partial \lambda_m^2}\right|_{\lambda_m=0} \lambda_m - 2c_2\lambda_m - c_1 sign(\lambda_m) \tag{5}$$

with an error in the λ_m^3 magnitude. Fig. 2 sketches two cases of the gradient of the objective function L with and without L1 regularization. In Fig. 2(a) the Rprop algorithm will set λ to 0 after an infinite number of iterations, while in Fig. 2(b) it will trim λ at some point > 0. We want to modify Rprop, so that it is able to distinguish these two cases already in one iteration. Without loss of generality we suppose $\lambda_{m,i} < 0$ (point (0) in Fig. 2). An update $s_{m,i} < 0$ keeps λ_m in the same orthant, but with $s_{m,i} > 0$ we have three cases: (1) we have case Fig. 2(a) where the sign of the gradient is changed, (2) we have case Fig. 2(b) and the gradient is not changed, and (3) the same case with a changed gradient. Case (1) and (3) can be discriminated by evaluating the expression $(\partial L/\partial \lambda_m)^2 - c_1^2$. So we propose to check in each iteration for each λ_m the questions

$$c_1^2 > (\partial L/\partial \lambda_m + c_1 sign(\lambda_m))^2 \approx L_0 \tag{6}$$

$$0 > \left\{\frac{\partial L}{\partial \lambda_m}\right\}_{i-1} \cdot \left\{\frac{\partial L}{\partial \lambda_m}\right\}_i \tag{7}$$

$$0 > \lambda_{m,i} \cdot \lambda_{m,i-1} \tag{8}$$

If they are true, set $\lambda_m = 0$ and skip the if clauses in the Rprop algorithm. At $\lambda_{m,i} == 0$ the evaluation of Eq. 6 is sufficient.

The modification in the last paragraph still needs the gradient of all λ_m, but experiments showed that the set of features bound by this approach to zero are seldom changed between iterations, so it is enough to check each feature only each Nth iteration. We propose to combine count cut-offs (only use features seen at least n-times in training corpus) and Elastic-Net, by evaluating features above the count cut-off every iteration and cut-off features only every Nth iteration. The sets of cut features which is evaluated rotates through λ_1^M by modulo. In Sec. 5, experimental results are presented.

Table 1. Effect of various feature-reduction techniques (up to 1%) on the performance. Elastic-Net (EN) uses $c_1 = 2^{-4}$.

feature set	#features	PER[%]		WER[%]	
		Dev	Eva	Dev	Eva
full model	54.603.236	7.7	7.9	33.8	34.2
+margin	54.597.879	7.4	7.9	32.3	34.2
elastic net (EN)	322.248	7.5	8.0	33.4	34.3
count > 0	802.379	7.9	8.1	34.1	34.7
rotating EN	260.658	7.7	8.0	34.6	34.6

4. LM IN CRF-SEARCH

One of the most powerful features of CRFs is the context feature on target side, the so-called bigram feature. It is computationally expensive, since the complexity of the model correlates with the context length, as already described in Sec. 1. Since longer context may lead to even better results but the complexity forbids the direct integration, one solution could be to integrate a classical language model (LM) into the CRF search process. This LM could easily be calculated beforehand and just be used as an additional knowledge source:

$$\hat{t}_1^N = \underset{t_1^N}{\operatorname{argmax}} \left\{ exp(H(t_1^N, s_1^N))^{1-\alpha} \cdot p_{LM}^{\alpha}(t_1^N) \right\} \qquad (9)$$

The LM is weighted using α.

The SRI LM Toolkit has been used to train the language models [7]. Experimental results are reported in the next section.

5. EXPERIMENTAL RESULTS

To get an idea of the effect of the just introduced methods, we performed a number of experiments on the NETtalk 15k corpus. This English g2p corpus is comprised of roughly 15k sentences for training, whereof 1k is set aside for development. The test set contains approximately 5k sentences. Since a large number of experiments with a detailed analysis had to be performed, such a small corpus is well suited. As error measure, we use phoneme error rate (PER) as well as word error rate (WER). For all experiments reported in this paper, we used the manual alignment which is available with the corpus.

We first optimized a baseline system on the corpus. Therefor, we checked a number of different features and came up with the following setup leading to our best result: lexical features in a window of $[-4, \ldots, 4]$ around the current word, i.e. at nine positions, the bigram feature and combined features. The latter features are composed of all monotone and overlapping combinations of lexical features of lengths two up to six.

For application of Elastic-Nets (EN) on this setup, the regularization Parameters c_1 and c_2 were re-tuned. We first justified c_2 with respect to PER on the development set. Afterwards, c_1 was tuned resulting in the interdependence of PER and number of features shown in Fig. 3. The number of features can be greatly reduced by up to two magnitudes without

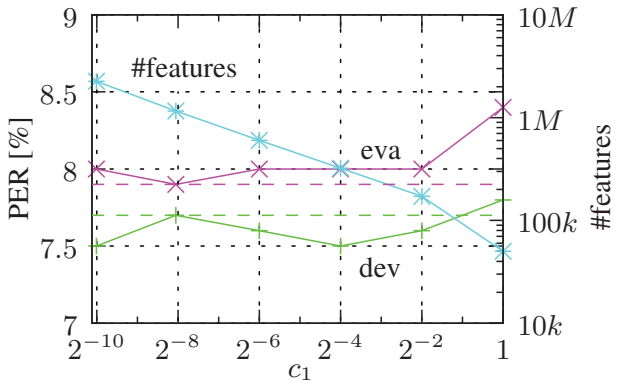

Fig. 3. PER on development and evaluation set vs. L1-regularization c_1. L2-reg. was kept fixed to $c_2 = 2^{-3}$. The dotted lines symbolize the PER without L1-reg.

loss of performance. The resulting feature sets are not equal to feature count cut-offs. 558k of 56M features were set to zero despite their count in the training corpus was not zero and 77k of 56M features were used though their count was zero. Tab. 1 documents the comparison of a model without feature selection, EN, count cut-offs and our proposed combination of count cut-offs and EN which we call rotating EN. It turned out that all feature selection methods reduced the number of features to less than 1%. There is no significant difference in performance , although rotating EN reduced the number of features the most.

The effect of a language model integrated into CRF search has been measured in the following way: first, standard ARPA LMs have been trained on the target side of the training corpus for the orders two up to seven. The lowest perplexity on the dev set is obtained for order 5 with 8.3.

We wanted to separate the effect of the various kinds of features and the language model. Thus, we trained six different systems, each incorporating different features. Tab. 2 gives an overview of the selected features. To each of the systems, the language models of order two up until seven have been combined. Therefor, for each experiment, the interpolation weight α had to be adjusted. We did this by grid search. The order and weighting factor of the best performing LM are presented in Tab. 2. The results are grouped according to the used features. Each experiment is reported with and without LM in search. If we have a look at the experiments where no combined features are incorporated, we can see that the LM can improve the system, even if bigram dependencies are considered within the CRF (third of the six experiments within the table). As soon as combined features are incorporated, the quality of the model greatly improves (even more than with the bigram features alone). Here, the additional LM can not improve the best performing system significantly, but it can in someway compensate for the bigram feature, since the result of the model with only the unigram feature can be improved with a 5-gram LM to give the same performance as with the bigram feature. This is also true, if the elastic net is applied. Thus, it would be possible to omit the bigram feature

Table 2. Results for language models integrated into CRF search. Additionally, the best LM n-gram order as well as the interpolation weight α is given. Tuning of LM for Experiment "*" is documented in Fig. 4. h_0 and h_1 represent the unigram and bigram feature respectively, lexical features are $(t_n = t', s_{n+\epsilon} = s')$, and source n-grams are combinations of successional lexical features. *EN* marks the experiment using Elastic Net in combination with the language model.

	feature set		LM	PER[%]		WER[%]		LM
				Dev	Eva	Dev	Eva	order / α
source lexicals + source n-grams				14.6	14.6	59.7	57.5	–
			✓	11.5	11.8	47.2	46.7	7 / 0.35
	$+h_0$			14.8	14.6	60.2	57.6	–
			✓	11.7	11.9	47.0	47.0	7 / 0.40
	$+h_1$			12.3	12.2	51.9	49.5	–
			✓	11.1	11.1	45.5	44.8	6 / 0.30
				8.0	8.3	35.4	35.9	–
			✓	7.4	8.3	33.0	35.8	6 / 0.20
	$+h_0$	*		8.0	8.3	35.0	36.0	–
			✓	7.4	7.9	32.6	34.5	4 / 0.25
		EN		7.9	8.4	35.7	36.5	–
			✓	7.5	7.9	33.8	34.2	4 / 0.25
	$+h_1$			7.4	7.9	32.3	34.2	–
			✓	7.3	7.8	32.1	33.5	5 / 0.10

and decrease training time (about 10%) and memory requirements greatly. Note that the results in Tab. 2 are roughly 6% better than those reported in [8], the most current publication using exactly this corpus. This can to some degree explained with the fact that discriminative models usually work better than generative ones if little training data is available.

6. CONCLUSION

In this paper, extensions to the popular CRF approach for NLP tasks have been proposed. Beside remarks on some small tweaks, like the use of the fast and easy to implement Rprop as optimization algorithm instead of the popular L-BFGS or the introduction of a margin into the training criterion, which significantly improves the systems, two approaches have been implemented. On the one hand, a very effective method of feature reduction has been ported to Rprop, reducing the size of the feature set to 1% while keeping the same performance. On the other hand, the incorporation of a standard LM on target side into the CRF search process has been tested. The idea was to take larger contexts than bigram dependencies into account. LMs could not improve the best system significantly, but the expensive bigram feature can in some way be compensated. I.e., if the LM is applied in search instead of the costly bigram feature, the results are comparable, even if additionally the elastic net is applied.

7. REFERENCES

[1] Thomas Lavergne, Olivier Cappé, and François Yvon, "Practical Very Large Scale CRFs," in *Proceedings the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*. July 2010, pp. 504–513, Association for Computational Linguistics.

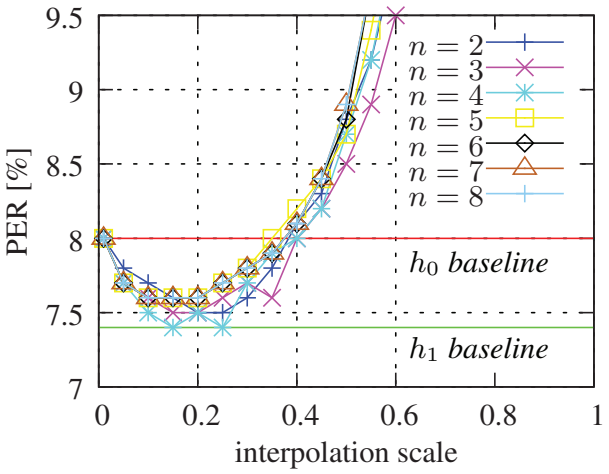

Fig. 4. Effect of the LM on the performance (PER on the development set) of the CRF system ("*" in Tab. 2) for various interpolation scales. The baseline feature set here includes lexical and combined features as well as a unigram feature.

[2] Thomas Deselaers, Saša Hasan, Oliver Bender, and Hermann Ney, "A Deep Learning Approach to Machine Transliteration," in *Proceedings of the EACL 2009 Workshop on Statistical Machine Translation*, Athens, Greece, Mar. 2009, pp. 233–241.

[3] Stefan Hahn, Patrick Lehnen, Georg Heigold, and Hermann Ney, "Optimizing CRFs for SLU Tasks in Various Languages Using Modified Training Criteria," in *Proceedings of ISCA Interspeech*, Brighton, U.K., Sept. 2009, pp. 2727–2730.

[4] M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: The Rprop algorithm," in *IEEE International Conference on Neural Networks (ICNN)*, San Francisco, CA, USA, March – April 1993, pp. 586 – 591.

[5] G. Heigold, R. Schlüter, and H. Ney, "Modified MPE/MMI in a Transducer-Based Framework," in *Proceedings of the Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Taipei, Taiwan, Apr. 2009.

[6] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proceedings of the Eighteenth International Conference on Machine Learning (ICML)*, Williamstown, MA, USA, June 2001, pp. 282–289.

[7] A. Stolcke, "SRILM - An Extensible Language Modeling Toolkit," Denver, CO, USA, Sept. 2002, pp. 901–904.

[8] Maximilian Bisani and Hermann Ney, "Joint-sequence models for grapheme-to-phoneme conversion," *Speech Communication*, vol. 50, no. 5, pp. 434–451, May 2008.