Topological Fields Chunking for German with SVM's: Optimizing SVM-parameters with GA's

Martina Liepert*
Seminar für Sprachwissenschaft
Universität Tübingen
Hausserstr. 11
72076 Tübingen, Germany
liepert@sfs.uni-tuebingen.de

Abstract

Support Vector Machines (SVM's) are used to chunk topological fields in German, a multi-class classification problem on highly unbalanced data. As SVM's are originally developed for binary classification we have to extend them to our multi-class problem. The negative effects of unbalanced data are compensated by introducing error penalties. Yet, it is unclear how to optimize these parameters. We compare several optimization approaches, amongst them optimization of SVM-parameters with genetic algorithms (GA's).

1 Introduction

Topological fields define parts within the German sentence which determine its basic structure of main- and subclauses. Once topological fields are found, it is much easier to perform bottom-up parsing steps like complex NP-chunking or to find grammatical relations. We do topological fields chunking for German with Support Vector Machines (SVM's). This task presents itself as a 4-class classification problem on highly unbalanced data. We solve the multi-class task by a divide-and-conquer-strategy (one-vs-one). To achieve good results, we have to optimize the hyperparameters of the SVM (γ and C's). We use Genetic Algorithms (GA's) for this optimization task and compare three different approaches: one where all C's and γ's are the same for all binary learners, one where the C's are chosen differently in each binary learner and one experiment where both, C and γ, are different in the binary classifiers. The best results achieved with the SVM are compared to corresponding results obtained with a memory based learner.

2 Support Vector Machines

Support Vector Machines (SVM's)[1] (Vapnik 98) are binary classifiers which try to separate two classes with a linear hyperplane, which is chosen such that the distance to the nearest training points becomes maximal. A Kernel maps the data into a higher dimensional feature space where it is more probably linear separable. For reasons of brevity we will not further introduce SVM's, please see e.g. (Schölkopf & Smola 02) for an excellent introduction. In our experiments we will focus on the RBF-Kernel with the Kernel-parameter γ. As the data is not linearly separable, we use the soft margin approach, where a slack variable ϵ is introduced and weighed by a constant C which penalizes training errors. A further extension of this approach, necessary especially with unbalanced data, is to weigh training errors of the two relevant classes differently by assigning a weight C_+ to the positive class and a weight C_- to the negative class.

Although some approaches have been presented to solve multi-class problems in one step ((Weston & Watkins 98), (Lee *et al.* 01)), we chose to solve the multi-class classification task by decomposing the problem into several binary problems (one-vs-one) and combining their results by majority voting. This approach is computationally less time consuming, easy to implement and shows comparable results (Hsu & Lin 01).

How to optimize the C-parameters and γ is an item of ongoing research. Heuristics as proposed in (Chew *et al.* 00) do not lead to the best possible output. Even if we find the optimal weighting for all single binary classifiers it might well be the case that this is not the best overall weighting for the combined multi-class classifier. A brute force search over a variety of weighting combinations would quickly lead to an exponentially growing search space and computation would soon be too time consuming. [2]

* This work has been supported by the German Research Council (DFG) as part of the Sonderforschungsbereich 441 "Linguistische Datenstrukturen". It was done during a stay of the author at the CNTS at the University of Antwerp. Many thanks to Walter Daelemans and his colleagues and special thanks to Boris Terzic, Jorn Veenstra and Tylman Ule.

[1] For all our experiments described in this paper we used the freely available SVM implementation LibSVM, version 2.36 (Chang & Lin 01), see http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[2] With 4 target classes (as in our data) we would have to train 6 binary classifiers. With 10 weighting combinations for each of them, we would have to test 10^6 combinations. The search space becomes even bigger if we take into account the search for an optimal γ in the RBF-kernel of each of the binary classifiers.

3 Genetic Algorithms

Genetic algorithms (GA's)[3] try to find an optimal solution through an evolutionary process over big search spaces. They start with a randomly created set of possible solutions (*chromosomes*). Using a fitness function, the GA calculates the fitness of each chromosome in the initial set and builds a first population by recombining parental chromosomes (*crossover*) and performing random changes (*mutation*). The best (fittest) chromosomes out of this population are used as offspring and will form the basis of the next generation. This process is repeated for a fixed number of generations or until another stopping criterion is fullfilled.

In our experiments, performed on a cluster of 28 computers, we chose 20 as the population size and 50 as the number of offspring. Crossover is done by uniform crossover which means that the bits from both parents are randomly copied. The crossover rate is 0.6. Mutation rate for C and for γ is 0.05. The fitness function in our experiments is the $F_{\beta=1}$ score[4] (Van Rijsbergen 79).

4 Topological Fields Chunking for German

4.1 Topological Fields in German

The theory of topological fields for German (Höhle 85) describes constituents of a sentence which are defined by the finite verb (the left bracket; LK), the infinite verbal complex (VC) and the subordinator in subclasses (C-field). For each of the three clause types in German (verb first - V1, verb second - V2, verb last - VL), Höhle defines a topological field model in which the sentence brackets LK, VC and C separate the section preceeding the LK (Vorfeld; initial field; VF), the part of the sentence included by the sentence brackets (Mittelfeld; middle field; MF) and the section following the VC (Nachfeld; final field; NF). In German, the ordering of the sentence brackets and of the initial field, the middle field and the final field is clearly syntactically restricted, whereas the position of other constituents in the German sentence (like verbal complements) is relatively free. Once the topological field structure of a sentence is known, we have the boundaries of all subclauses in the sentence and can investigate these smaller structures in a divide and conquer strategy. It is then much easier to find e.g. the verbal complements and the number of possible ambiguities is considerably reduced. Moreover the distribution of many syntactic phenomena is strongly dependent on the topological fields. Thus the annotation with topological fields is a top-down step which simplifies following bottom-up parsing steps considerably. A simple illustrative sentence can be found in Fig. 1.

4.2 Previous Work on Topological Fields Chunking for German

(Veenstra *et al.* 02) have compared three different approaches to topological fields chunking. They used Finite State Automata (FSA), probabilistic context free grammars (PCFG) and a memory-based learner (MBL). The results of Veenstra *et al.* are given in Table 1.

4.3 Data

The TübaD/Z is a database of currently ca. 9000 hand annotated sentences taken from the taz newspaper corpus. The annotation scheme of the TübaD/Z mainly follows the Verbmobil standard (Stegmann *et al.* 98). One of our aims was to compare our results with the previously achieved ones, so we chose the same training data as Veenstra et al. (4523 sentences from the TübaD/Z data base). We used their test set, consisting of 1613 sentences, as validation set and 4594 sentences as test set on which we tested the SVM models as well as the memory based learner from (Veenstra *et al.* 02). We performed our experiments on TnT-tagged data (Brants 00) from the TübaD/Z. This tagger has an overall accuracy of 94.7% on the TübaD/Z. Following (Ramshaw & Marcus 95), we represented our data in the IOB-format which results in a multiclass learning problem with four target classes (O, I-VC, I-C, I-LK). These four target classes are distributed over the training data as follows: 80 640 training points in class O, 6759 in class VC, 2251 in class C and 5827 in class LK, thus presenting a learning problem on highly unbalanced data.

	ALL	LK	VC	C
FSA	94.1	96.2	92.0	93.8
PCFG	94.4	97.0	92.2	92.3
MBL	93.3	96.0	90.0	91.6
baseline	75.5	75.2	72.3	83.2

Table 1: Results ($F_{\beta=1}$ score) of Veenstra *et al.*

[3] As basis of the GA described in this paper we used a μ, λ-EA-implementation done by Bart Naudts, Intelligent Systems Lab, Department of Mathematics and Computer Science, University of Antwerp, Belgium.

[4] In order to calculate the $F_{\beta=1}$ score we used the CoNLLeval-script written by Erik Tjong Kim Sang which is freely available at http://cnts.uia.ac.be/conll2000/chunking/conlleval.txt.

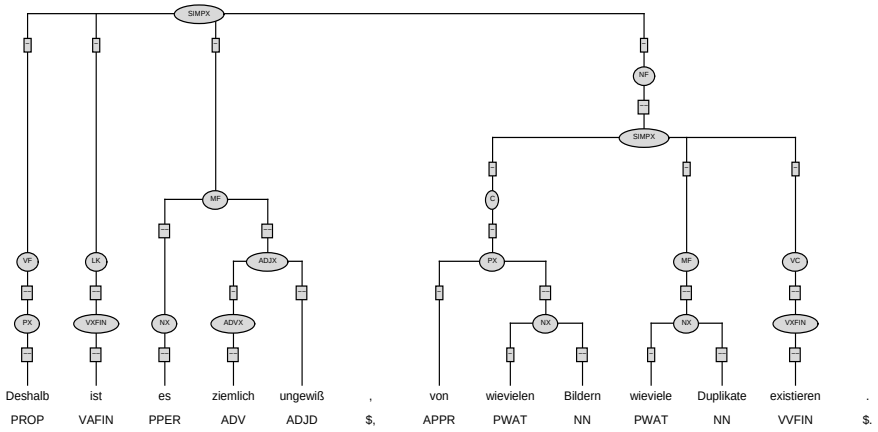

Figure 1: Sentence from the TübaD/Z: ”Hence, it is quite uncertain, of how many pictures there exist how many duplicates.” (LK: ”ist”; C: ”von wievielen Bildern”; VC: ”existieren”)

5 Experiments

5.1 Features and Data Representation

Following (Veenstra *et al.* 02), we used the window approach (2 left, 1 right) and POS-information as features (55 possible POS-tags). For LibSVM we encoded the features into a 220 dimensional (sparse) vector.

5.2 Baseline

As a baseline we assigned each instance its most frequent tag and obtained the results on the evaluation set as shown in Table 1 and on the test set as shown in Table 5. Additionally we trained LibSVM with the default-values (C = 1, $\gamma = \frac{1}{220}$) and tested it on the test set (see Table 5).

5.3 LibSVM Model Selection Tool

The LibSVM package contains an easy to use model selection tool that offers the possibility to find the best C and γ parameter combination for SV classification. The user can determine the range of C and γ values[5]. The tool will then train a model with each combination of the given parameters. For each model it will output the accuracy calculated with cross validation techniques. LibSVM handles multi-class classification problems with the same approach we chose (one-vs-one with majority vote). But note that LibSVM does not use class (and binary classifier) specific C and γ values but the same C and γ-values for both classes in all binary classifiers..

We ran the LibSVM model selection tool on our training data with 5-fold cross validation and chose the values -1 ... 5 for C, and the values -4 ... 1 for γ. The best among the 42 calculated models was the one with the value 8 for C and the value 0.0625 for γ in LibSVM. This model got a result of 92.14 as $F_{\beta=1}$-score on the test set (Table 5).

[5]In the model selection tool, values x are chosen $x \in \mathbb{Z}$; the values which are used as parameters in LibSVM are then 2^x.

5.4 Optimizing C-values

In these experiments we chose a fixed γ and tried to find the best class specific C-values for each binary classifier by using a GA as presented in Sect. 3. The chromosomes in the GA consist of integer arrays of length 12; each position indicating the C_+ respectively C_- value for one of the binary classifiers. With regard to the number of instances in the single target classes, we set the range of possible C-values to [0 ... 40]. In order to save training time we decided to reduce the training set for the GA to 65000 instances according to a learning curve we produced with the parameters C = 8 and g = 0.0625 (Fig. 2). With the best parameter combination found by the GA on this reduced training set, we then trained a model on the whole training set and tested it on the test set. According to experiences with previous experiments on the same data we chose the following values for γ: 0.0625, 0.125, 0.25, 0.5 and 1. The results obtained in each generation on the reduced training set can be seen in Table 2. The best result was achieved with $\gamma = 0.0625$. We present an optimal chromosome in Table 3. The corresponding model (trained on the complete training set and tested on the test set) got an $F_{\beta=1}$-score of 92.25 (see Table 5).

	I-LK/O	I-LK/I-VC	I-LK/I-C	O/I-VC	O/I-C	I-VC/I-C
C_1/C_2	14/13	10/19	20/18	15/27	16/11	23/10

Table 3: Optimizing C: best chromosome after 15 generations ($\gamma = 0.0625$). The first row shows which classes the respective binary classifier focusses on. C_1 refers to the first mentioned target class, C_2 to the latter.

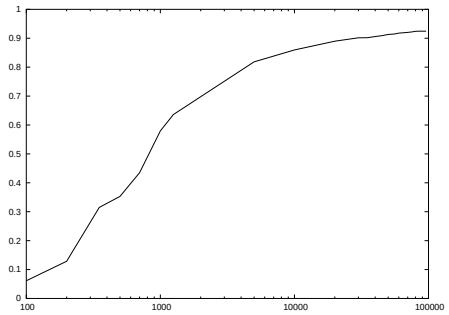

Figure 2: Learning curve trained with LibSVM

gen.	0.0625	0.125	0.25	0.5	1
1	92.90	92.83	92.27	91.80	87.90
2	92.96	92.96	92.40	91.93	87.97
3	93.00	92.96	92.48	91.99	87.98
4	93.08	92.96	92.48	91.99	87.98
5	93.13	92.96	92.51	91.99	88.00
6	93.16	92.97	92.52	92.01	88.00
7	93.16	92.97	92.54	92.01	88.00
8	93.17	92.98	92.54	92.01	88.02
9	93.18	92.98	92.60	92.03	88.02
10	93.18	92.98	92.61	92.03	88.02
11	93.18	92.98	92.61	92.03	88.03
12	93.19	92.98	92.61	92.03	88.04
13	93.19	92.98	92.61	92.03	88.04
14	93.19	92.98	92.61	92.04	88.04
15	93.19	92.98	92.61	92.04	88.04

Table 2: Optimizing C: results of the GA.

5.5 Optimizing C's and γ's

In the experiments described in this section not only the class specific C-values were to be optimized but also the γ-values. Thus the γ-values of the different binary learners may differ from each other. Correspondingly the chromosomes consist of integer arrays of length 18: 12 positions for the C-values and 6 positions for the γ-values of the binary classifiers. The γ-value is encoded such that the integer value in the chromosome has to be multiplied with 0.05 to give the final value. The possible range for γ is [0.05 ... 1.5]. Again, the GA was trained on the reduced training set and tested on the evaluation set. As the chromosomes are significantly longer than in the previous experiments, more generations were necessary until the GA converged. The results of the first 30 generations can be seen in Fig. 3; the best score was 93.09. C and γ-

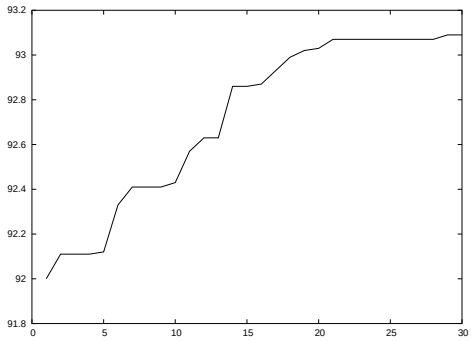

Figure 3: Optimizing C and γ: Results of the GA.

values corresponding to an optimal chromosome are presented in Table 4. With this parameter combination we obtained a score of 92.25 on the test set (see Table 5).

5.6 Results

We trained the classifiers with the best results from Sect. 5.3, Sect. 5.4 and Sect. 5.5 on the complete training set and tested these models on the test data. Additionaly we tested the MBL on the same data. For the MBL we used the decision tree variant (IGTree) of Timbl (Daelemans *et al.* 01) with information gain as feature weighting method and 1 as the number of nearest neighbours. The results are reported in Table 5.

6 Comparison and Discussion

6.1 Comparison of Our Results

The classifier trained with the hyperparameters found by the model selection tool, thus with the same C- and γ- values for both classes in all binary classifiers, reached a surprisingly high score of 92.14. The

	ALL	LK	VC	C
model selection	92.14	94.40	90.82	88.71
optimizing C	92.25	94.62	90.72	89.10
opt. C and γ	92.25	94.62	90.85	89.10
MBL	92.94	95.57	91.66	88.24
LibSVM default	77.74	78.85	76.66	77.35
baseline	75.4	81.1	62.4	81.9

Table 5: Results on the test set

	I-LK/O	I-LK/I-VC	I-LK/I-C	O/I-VC	O/I-C	I-VC/I-C
C_1/C_2	8/12	14/23	11/22	22/36	6/10	11/32
γ	0.1	0.05	0.1	0.05	0.2	0.3

Table 4: Optimizing C and γ: best C- and γ-combination after 30 generations.

optimization with the GA in 5.4 could only outperform this score by 0.11. Optimizing both, C's and γ's, for all binary classifiers could not further improve the score. No doubt, optimization of the training error penalty parameter and γ is important (cf. the results with LibSVM default values in Table 5). Yet, varying γ for each binary classifier does not seem to be advantageous.

None of the learners in our experiments were able to outperform the MBL which scored 92.94 on the test data. It seems that this classifier is slightly better suited to learn topological fields with features as chosen in our experiments.[6]

The best results in Sect. 5.4 were found with $\gamma = 0.0625$, a small value for γ which could imply the danger of overfitting. The average γ-value in the best chromosome found in 5.5 is ca. 3 times bigger, yet, this classifier did not score better on the test set. Another interesting point is that the optimal C-values found in 5.4 and 5.5 do not conform with the heuristics proposed in (Chew *et al.* 00) nor with the number of training points in the single target classes. Thus, heuristics can hardly help to find the best parameter values.

6.2 Comparison with Previous Work

Very recently, Xu and Chan have also performed SVM optimization experiments using GA's. In (Xu & Chan 03b), they used GA's in order to optimize class specific C-values with a fixed γ. In (Xu & Chan 03a), they optimized γ for fixed C-values. In these experiments they could improve accuracy by 0.4% and 0.25%. However, they did not compare both approaches on the same data, nor did they include experiments corresponding to ours in 5.3 and 5.5. This may be the reason why their conclusion is slighty different from ours: they seem to propose that optimizing classifier specific γ's and class specific C's with GA's leads to considerably better results. On our data though, the model selection tool with the same C- and γ-values for all binary classifiers got competitive results as well.

[6]The differences between the best scores of the SVM on the validation st and the MBL in (Veenstra *et al.* 02) are much smaller (0.11 and 0.21). Yet, it is difficult to compare these scores as the same data was used as test data for the MBL but as validation set for the SVM's.

7 Future Work

It would be interesting to experiment with different kinds of data representation or to focus on feature selection, e.g. to use the previously predicted chunk in the same sentence as an additional feature. Regarding SVM's a next step could be to use different Kernels. Concerning the GA's it would be interesting to use different GA implementations and search strategies. We will concentrate on the first mentioned direction.

References

(Brants 00) T. Brants. Tnt – a statistical part-of-speech tagger, 2000.

(Chang & Lin 01) C. Chang and C. Lin. Libsvm: a library for support vector machines, 2001.

(Chew *et al.* 00) H.G. Chew, R.E. Bogner, and C.C. Lim. Target detection in radar imagery using support vector machines with training size biasing. In *International Conference on Control, Automation, Robotics and Vision, ICARCV 2000, Singapore*, pages CD–ROM, 2000.

(Daelemans *et al.* 01) Walter Daelemans, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch. Timbl: Tilburg memory-based learner - version 4.0 reference guide, 2001.

(Höhle 85) Tilman Höhle. Der Begriff 'Mittelfeld'. Anmerkungen über die Theorie der topologischen Felder. In *Akten des VII. Internationalen Germanisten-Kongresses*, pages 329 – 340, Göttingen, 1985.

(Hsu & Lin 01) C.W. Hsu and C.J. Lin. A comparison on methods for multi-class support vector machines. Technical report, Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, 2001.

(Lee *et al.* 01) Y. Lee, Y. Lin, and G. Wahba. Multicategory support vector machines. Technical Report 1043, Department of Statistics, University of Wisconsin, Madison WI, 2001. Proceedings of the 33rd Symposium on the Interface, 2001.

(Ramshaw & Marcus 95) Lance Ramshaw and Mitch Marcus. Text chunking using transformation-based learning. In David Yarovsky and Kenneth Church, editors, *Proceedings of the Third Workshop on Very Large Corpora*, pages 82–94, Somerset, New Jersey, 1995. Association for Computational Linguistics.

(Schölkopf & Smola 02) B. Schölkopf and A.J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

(Stegmann *et al.* 98) Rosmary Stegmann, Heike Telljohann, and E. W. Hinrichs. Stylebook for the german treebank in verb-mobil. technical report, 1998.

(Van Rijsbergen 79) C. J. Van Rijsbergen. *Information Retrieval, 2nd edition.* Dept. of Computer Science, University of Glasgow, 1979.

(Vapnik 98) V. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

(Veenstra *et al.* 02) Jorn Veenstra, Frank Henrik Müller, and Tylman Ule. Topological fields chunking for german. In *Proceedings of the Sixth Conference on Natural Language Learning (CoNLL 2002)*, pages 56–6, Taipei, Taiwan, 2002.

(Weston & Watkins 98) J. Weston and C. Watkins. Multi-class support vector machines, 1998.

(Xu & Chan 03a) Peng Xu and Andrew K. Chan. An efficient algorithm on multi-class support vector machine model selection. In *Proceedings of the International Joint Conference on Neural Networks 2003*, 2003.

(Xu & Chan 03b) Peng Xu and Andrew K. Chan. Support vector machines for multi-class signal classification with unbalanced samples. In *Proceedings of the International Joint Conference on Neural Networks 2003*, 2003.