A POS Tagger for Social Media Texts Trained on Web Comments

Melanie Neunerdt, Michael Reyer, and Rudolf Mathar

Abstract—**Using social media tools such as blogs and forums have become more and more popular in recent years. Hence, a huge collection of social media texts from different communities is available for accessing user opinions, e.g., for marketing studies or acceptance research. Typically, methods from *Natural Language Processing* are applied to social media texts to automatically recognize user opinions. A fundamental component of the linguistic pipeline in *Natural Language Processing* is *Part-of-Speech* tagging. Most state-of-the-art *Part-of-Speech* taggers are trained on newspaper corpora, which differ in many ways from non-standardized social media text. Hence, applying common taggers to such texts results in performance degradation. In this paper, we present extensions to a basic Markov model tagger for the annotation of social media texts. Considering the German standard *Stuttgart/Tübinger TagSet* (STTS), we distinguish 54 tag classes. Applying our approach improves the tagging accuracy for social media texts considerably, when we train our model on a combination of annotated texts from newspapers and Web comments.**

Index Terms—**Natural language processing, part-of-speech tagging, opinion mining, German.**

I. Introduction

SOCIAL media applications lead to constantly growing user generated content in the Web. Different types of social media tools, e.g., blogs, forums as well as news sites allow users to post Web comments. This kind of consumer-to-consumer communication can efficiently be used to access user opinions for marketing studies or acceptance research. A beneficial property of Web comments is the fact that the data is natural and authentic, and public. Furthermore, opinions from proponents, opponents as well as from impartial persons can be obtained from different Web domains, i.e., different communities.

Besides automatic opinion recognition, many other *Natural Language Processing* (NLP) methods such as syntactical parsing, machine translation or text summarization require *Part-of-Speech* (POS) tag information for a given word sequence. State-of-the-art POS taggers are basically developed for the task of tagging standardized texts such as newspaper articles which are grammatically approved. Hence, parameter estimation is usually performed on newspaper text corpora as training data. Web comments, however, differ from standardized text, since they are characterized by a spoken language, a dialogic and an informal writing style. This poses some special challenges to deal with in developing methods for automatic POS tagging of Web comments. These are particularly, the treatment of unknown (out-of-vocabulary) words and the different grammatical structure of social media texts in contrast to newspaper text. Furthermore, text genre specific manually annotated corpora, i.e., Web comments are required for training and testing. To the best of our knowledge all large manually annotated corpora are exclusively newspaper texts.

In this work, we propose a Markov model tagger with parameter estimation enhancements for the POS annotation of social media texts. We apply and evaluate the tagger for German social media texts exemplarily. In order to make our method usable for NLP methods requiring POS information, e.g., syntactical parsing, we use the 54 *Stuttgart/Tübinger TagSet* (STTS) tag classes without any text genre specific extensions. Web comments are not completely different from newspaper texts. However, due to the dialogic text style, i.e. the distribution of POS sequences changes, the grammatical structure differs. Hence, the training is performed on a combination of newspaper and Web comment corpora. Results are compared to state-of-the-art/widely used POS taggers. We particularly study the influence of additional Web comment training data for our approach and compare results to those achieved by methods basically developed for standardized texts. The proposed approach outperforms state-of-the-art POS taggers significantly for German social media texts.

The outline of this paper is as follows: Section II summarizes the related work to provide an overview of POS tagging. In Section III we introduce *WebTagger* for automatic POS tagging applied to social media texts. Section IV presents experimental results considering different aspects, particularly discussing the adaptability to other languages in Subsection IV-E. Section V covers the conclusion and discusses future work.

II. Related work

Different statistical approaches have been proposed to solve the task of automatic POS tagging. Typically, POS taggers utilize two different probabilistic models, a Maximum entropy model or a Markov model capturing lexical and contextual information. Common Maximum entropy based taggers are proposed in [1], [2]. These approaches are adapted by using

Manuscript received on August 2, 2013; accepted for publication on September 30, 2013.

The authors are with the Institute for Theoretical Information Technology, RWTH Aachen University, Germany (e-mail: {neunerdt, reyer, mathar}@ti.rwth-aachen.de).

different features for the model. Toutanova et al. [1] propose the Stanford tagger modeling the sequence of words as bi-directional dependency network considering lexical and tag context information. Markov model taggers are proposed in [3], [4], [5]. TreeTagger [4] and TnT [5] use a second order Markov model applying some smoothing techniques for the estimation of lexical probabilities. The Stanford tagger and the TreeTagger are trained on corpora in different languages, which shows their generality in application. Both taggers use the STTS [6] tag set for German, which is commonly used for NLP methods. Furthermore, some other machine learning techniques, e.g., Support Vector Machines [7] and Neural Networks [8], are applied to the problem of automatic POS tagging.

In [9], [10] different POS taggers are compared and evaluated for German. Schneider et al. [9] point out the performance loss on unknown words, of a rule-based tagger compared to a statistical tagger. Five state-of-the-art taggers applied to Web texts are studied in [10]. Accuracy drops significantly for different Web text genres. Hence, the automatic annotation of Web texts is not yet a solved task.

Gimpel et al. [11], [12], [13] address the task of tagging non-standardized texts, characterized by a high number of unknown words. Gadde et al. [2] introduce adaptions to the Stanford tagger to handle noisy English text. Results are evaluated based on an SMS dataset. In [14] a twitter tagger based on a conditional random field (CRF) with features adapted to twitter characteristics is proposed. They propose some additional word clustering and further improvement to their method in [13].

III. WebTagger

WebTagger has much in common with the TreeTagger [4]. The taggers differ in the way the lexical probabilities are estimated, in particular for unknown words. We use the STTS tagset for annotation. Annotation rules for social media characteristics are given in [15], [16] and [17]. The general tagger model is taken from [4] and explained in Subsection III-A. Subsection III-B comprises our proposed enhancements to the basic tagger to improve POS tagging performance for social media texts. In contrast to other approaches estimation of lexical probabilities is extended by mapping unknown tokens (words), to tokens known from training or to some regular expressions. This mapping improves the estimation of lexical probabilities. Furthermore, prefix and suffix lexicon information are efficiently combined. Finally, in contrast to standard methods we suggest to use a semi-supervised auxiliary lexicon instead of information from automatically tagged or unsupervised training data.

A. Model

As tagger model we use an enhanced standard Markov model. In this subsection we explain the basic model. The aim of the tagger is to predict the associated POS tag sequence $t_1, \dots, t_n, \dots, t_N$ with $t_n \in \mathcal{T}$ (STTS) for a given sequence of tokens $w_1, \dots, w_n, \dots, w_N$ with $w_n \in \mathcal{W}$, where $\mathcal{W}$ contains all possible tokens. In order to achieve that the optimization problem

$$\hat{t}_1^N = \arg\max_{t_1^N} P(t_1^N, w_1^N)$$

using the notation for a sequence of POS tags t_l^n

$$t_l^n = \begin{cases} (t_l, \dots, t_n) & 1 \leq l \leq n \leq N \\ (t_1, \dots, t_n) & l \leq 0 \end{cases}$$

with $l \in \mathbb{Z}$, $n \in \mathbb{N}$, and $l \leq n \leq N$ is solved. The sequence of tokens w_l^n is defined analogously. This is a huge optimization problem which is simplified by the following approach. First, the probability chain rule for w_N and t_N to describe the joint probability by conditional probabilities is applied:

$$P(w_1^N, t_1^N) = \tag{1}$$
$$P(w_N \mid w_1{}^{N-1}, t_1^N) P(t_N \mid w_1^{N-1}, t_1^{N-1}) P(w_1^{N-1}, t_1^{N-1}).$$

As in [4] we use the assumptions

$$P(w_n \mid w_1^{n-1}, t_1^n) = P(w_n \mid t_n),$$
$$P(t_n \mid w_1^{n-1}, t_1^{n-1}) = P(t_n \mid t_{n-k}^{n-1})$$

with $k \in \mathbb{N}$. Applying those assumptions, a simple law of conditional probability, and iterating the procedure described in (1) leads to the equation:

$$P(w_1^N, t_1^N) = \prod_{n=1}^{N} \underbrace{\frac{P(t_n \mid w_n)}{P(t_n)}}_{\text{Lexical Prob.}} p(w_n) \underbrace{P(t_n \mid t_{n-k}^{n-1})}_{\text{Transition Prob.}}.$$

The assumptions are also referred to as k-order Markov model for transition probabilities and zero-order Markov model for the lexical probabilities. Moreover, the token probabilities $p(w_n)$ do not change with the tag sequences, and hence, may be omitted. Overall, this allows to model transition and lexical probabilities independently and the optimization task is rephrased as

$$\hat{t}_1^N = \arg\max_{t_1^N} \left\{ \prod_{n=1}^{N} \frac{P(t_n \mid w_n)}{P(t_n)} P(t_n \mid t_{n-k}^{n-1}) \right\}.$$

Before this optimization problem can be solved those probabilities have to be determined. The estimation of transition probabilities is taken from TreeTagger [4] by applying the ID3 algorithm [18]. Due to the ungrammaticalities, particularly given in social media texts, a high number of unseen contexts, e.g., trigrams, might occur when applying the tagger. In order to get reliable estimates in such cases, zero probabilities are replaced by a predefined small value. Furthermore, we propose to use manually annotated social media texts as additional training data, in order to learn different tag contexts given by the dialogic style characteristics of such texts. Lexical probabilities are estimated by our proposed methods, described in the following Subsection III-B.

B. Enhancements

In this subsection our enhancements to a basic Markov model tagger are introduced. This comprises the estimation of lexical probabilities, particularly for unknown tokens (words). Considering the high frequency of unknown tokens and their variability of POS classes makes the task of probability estimation more complex compared to newspaper texts. Hence, this problem can not be solved adequately by standard methods and more sophisticated methods are needed.

1) Token preprocessing: The given sequence of tokens usually contains tokens which do not occur in the training set. The preprocessing aims at mapping these tokens to related known tokens, if there exists a fitting token. Related tokens can be obtained by some transformations steps described by $t(w_n)$. These steps contain cross-language transformations as well as some transformations specifically for German. Amongst others character iteration correction, e.g., Hellooooooo→Hello, or *Umlaut* correction, e.g., Huette→Hütte (cottage), or character correction, e.g., Kuss→Kuß (kiss). Such transformations may be interpreted as substituting tokens by its normalized version. Therefore, we are calling this kind of transformation *normalization*. Furthermore, there are language independent *word classes* which are easily recognized and anticipated using regular expressions. Some examples are emoticons, e.g., *:-)* and *:(*, and URLs, e.g., http://www.test.de, xy.ch, multiple punctuation marks, e.g., and ....., *!!!*, number replacements, e.g., *50er*. The set of possible POS tags for each word class differs from one to three. In summary, our preprocessing step substitutes unknown tokens by its transformation, if this is within the training set, or returns the regular expression r, if the word is described by it or returns the marker for unknown tokens ϵ. This procedure is described by the mapping function $m : \mathcal{W} \to \mathcal{X} \cup \{\epsilon\}$ which is defined as

$$m(w_n) = \begin{cases} /r/ & w_n \in \mathcal{W}_r, \\ w_n & w_n \in \mathcal{L} \setminus \mathcal{R} \\ t(w_n) & t(w_n) \in \mathcal{L} \setminus \mathcal{R} \land w_n \notin \mathcal{L} \setminus \mathcal{R} \\ \epsilon & \text{elsewise.} \end{cases} \tag{2}$$

An overview of the corresponding word sets is given in Table I.

TABLE I
WORD SETS.

Word Set	Description
$\mathcal{R}$	Tokens covered by regular expressions
$\mathcal{L}$	Full form lexicon created from training data
$\mathcal{X} = \mathcal{L} \cup \mathcal{R}$	Full form lexicon extended by regular expressions
$\mathcal{W}$	All possible tokens

The word set $\mathcal{X}$ contains all tokens given by the full form lexicon $\mathcal{L}$ created from supervised training data, extended by the set of words $\mathcal{R}$ created by all regular expressions $r \in R$ as follows:

$$\mathcal{W}_r = \{w \in \mathcal{W} \mid w \sim /r/\}$$

indicates all tokens covered by a regular expression $r \in R$, thus $\mathcal{R} = \dot{\bigcup}_{r \in R} \mathcal{W}_r$.

2) Parameter estimation: Before our tagger can be used for predicting POS tag sequences, the probability parameter values have to be estimated. Therefore, we basically use a supervised learning approach, but extend this by some semi-supervised learning technique which is explained in more detail in the following section. A manually annotated training corpus

$$\mathcal{TR} = \left\{(\hat{w}_n, \hat{t}_n) \mid 1 \leq n \leq N\right\}$$

is used, where for each word $\hat{w}_n$ the correct tag $\hat{t}_n$ is known. The lexicon is given as

$$\mathcal{L} = \{\hat{w}_n \mid 1 \leq n \leq N\}\,.$$

We assume lexical probabilities to be position independent. Hence, we replace $P(t_n \mid w_n) = P(t \mid w)$ in the following notation. If the word $m(w)$ is known, i.e., it occurs in the training set $\mathcal{TR}$, the estimation is given by

$$\hat{P}_L(t \mid w) = \frac{\mid \{k \mid (\hat{t}_k, \hat{w}_k) = (t, w)\} \mid}{\mid \{k \mid \hat{w}_k = w\} \mid},$$

where the index L indicates that the word w is in the lexicon $\mathcal{L}$.

In the following the estimates if the word $m(w)$ is not in the lexicon $\mathcal{L}$ are described. First, we explain the estimation of the probabilities, if the unknown word is represented by a regular expression. The probabilities are given as

$$\hat{P}_R(t \mid r) = \frac{\mid \{k \mid \hat{t}_k = t \land \hat{w}_k \in \mathcal{W}_r)\} \mid}{\mid \{k \mid \hat{w}_k \in \mathcal{W}_r\} \mid}. \tag{3}$$

Using these estimates for regular expressions enables to assign reliable tag distributions even to previously unseen tokens from training.

Now we deal with (still) unknown tokens. From the training data set we determine all prefixes and suffixes of maximal length five. The description of all tokens having the same prefix/suffix may be realized with a regular expression. Hence, we assess the lexical probabilities for all given prefixes and suffixes as in (3). However, to improve the quality of our estimation we combine the probabilities for prefixes and suffixes as follows:

$$\hat{P}_{PS}(t \mid w) = \\ \frac{\mid \{k \mid \hat{t}_k = t \land \hat{w}_k \in \mathcal{W}_{p(w)})\} \mid + \mid \{k \mid \hat{t}_k = t \land \hat{w}_k \in \mathcal{W}_{s(w)})\} \mid}{\mid \{k \mid \hat{w}_k \in \mathcal{W}_{p(w)}\} \mid + \mid \{k \mid \hat{w}_k \in \mathcal{W}_{s(w)}\} \mid}$$

where $p(w)/s(w)$ are the regular expressions for the prefix/suffix of the word w. Common approaches use the joint probabilities of the independent prefix and suffix distributions. However, combining prefix and suffix lexical probabilities by the arithmetic mean, makes the method robust for uncommon prefix and/or suffix, which arise from informal writing style characteristics, e.g., word shortenings or typing errors. The proposed method improves tagging accuracy by 0.5 percentage points compared to the commonly used joint probability approach. In summary, the lexical probability is given as

$$P(t \mid w) = \begin{cases} \hat{P}_L(t \mid m(w)) & m(w) \in \mathcal{L}, \\ \hat{P}_R(t \mid m(w)) & m(w) \in \mathcal{R} \setminus \mathcal{L}, \\ \hat{P}_{PS}(t \mid w) & w \in (\mathcal{P} \cup \mathcal{S}) \setminus \mathcal{X}, \\ \hat{P}_S(t \mid w) & \text{elsewise}, \end{cases} \qquad (4)$$

where $\mathcal{P}/\mathcal{S}$ describes all tokens with known prefixes (suffixes). The last case in the description is by default given by the frequencies of the tags in the training set

$$\hat{P}_S(t \mid w) = \frac{|\{k \mid \hat{t}_k = t\}|}{N},$$

which is independent of the word w.

3) Semi-supervised learning: Preparing a fully supervised training text is a time-consuming job. In this subsection we propose an alternative approach which reduces the annotation effort considerably. The basic idea is as follows. The tagger is used for automatic tagging of a large social media text corpus.

$$\mathcal{SL} = \{(w_m, t_m) \mid 1 \leq m \leq M\}$$

The most frequent unknown tokens, $m(w_m) = \epsilon$, are determined and added to an auxiliary lexicon $\mathcal{L}^+$. For all tokens w_m of the auxiliary lexicon the possible tags are manually assigned and denoted by $\mathcal{T}_{w_m}$. If there is more than one tag possible, an adequate tag distribution needs to be assigned as well. Therefore, two approaches are utilized.

First, if at least one word $\hat{w}_k$ of the manually annotated training corpus has the same POS tag set as the manually assigned set $\mathcal{T}_{w_m}$, the cumulated tag distribution of those words is taken. Hence, the lexical probability is refined as

$$\hat{P}_{\mathcal{L}^+}(t \mid w_m) = \frac{|\{k \mid \hat{t}_k = t \wedge \mathcal{T}_{\hat{w}_k} = \mathcal{T}_{w_m})\}|}{|\{k \mid \mathcal{T}_{\hat{w}_k} = \mathcal{T}_{w_m}\}|},$$

where $\mathcal{T}_{\hat{w}_k} = \{\hat{t}_l \mid \hat{w}_l = \hat{w}_k\}$.

Second, if there is no word with the same set of possible tags in the training text, further manual annotation is performed. A reliable amount of such tokens is manually annotated in the large social media corpus SL. The resulting tag distribution is assigned to such unknown tokens.

Such unknown tokens are often shortened verbs or wrongly uncapitalized tokens. The following example illustrates that particular case. Consider the word *benutz* (*use*), which in a standardized text is just used as an imperative content word (VVIMP). In social media texts the word is also used as short form for the verb inflections *benutze* (*[I] use*) and *benutzt* (*[he] uses*). Hence, the resulting tag distribution from manual annotation of the large social media corpus results in $P(VVFIN) = 59\%, P(VVIMP) = 32\%$ and $P(VVPP) = 9\%$.

IV. Results

Different criteria are analyzed to evaluate the proposed approach. First, WebTagger is compared to four state-of-the-art POS taggers, considering German Web comments. Second, the performance improvements for each proposed enhancement step are demonstrated. Furthermore, the improvement by using manually annotated Web comments for training is pointed out. Particularly, we show that using non-standardized texts for training does not lead to a significant degradation when applying WebTagger to standardized newspaper texts. Finally, we study the transferability to different social media text types, where the taggers are not trained on the particular type.

A. Corpora

Two corpora are used for the purpose of training our new social media corpus *WebTrain* and a newspaper text corpus. *WebTrain* corpus contains 429 Web comments collected from *Heise.de*, which is a popular German newsticker site treating different technological topics. Each of 36,000 token is annotated with manually validated POS tags and lemmas. Annotation rules, particularly for social media text characteristics, are taken from [16]. The average POS tag ambiguity of tokens contained in the corpus is 2. This is significantly higher as the ambiguity in German newspaper texts, e.g., 1 for the *TIGER* corpus containing 890,000 tokens. In order to provide a sufficiently large training data amount, we combine *WebTrain* with the *TIGER* treebank [19] newspaper text corpus. We call that joint-domain training. *WebTrain* texts contain 18% trigrams, that never occur in the newspaper corpus *TIGER*. Those trigrams constitute 6% frequency of all *WebTrain* trigram counts. Both results motivate the need of text genre specific training data for reliable estimation of transition probabilities, e.g., for trigrams.

To have a deeper look in the general applicability of *WebTagger* for social media texts, an additional corpus *WebTypes* is used. It is composed of roughly 4,000 tokens, where comments from different Web sites and a corpus extract from the *Dortmunder chat corpus BalaCK 1-b* [20] are manually annotated. Three different types of social media texts are represented, YouTube comments, blog comments, and chat messages. Further corpus statistics can be found in [15].

B. Cross validation

A 10-fold cross validation is performed to evaluate the tagging accuracy of our approach, compared to state-of-the-art taggers. Therefore, *WebTrain* is divided into ten equally sized subsets which are created by randomly selected sentences. WebTagger and three state-of-the art taggers are trained on a combination of nine subsets and *TIGER* data in each validation step. The remaining subset is used for testing. The selected taggers are TreeTagger [4], TnT [5], and Stanford [1]. In a previous study [15] we evaluated the performance of the mentioned taggers for social media texts in more detail. Total tagging accuracies and accuracy rates achieved for known tokens and unknown tokens are determined. Mean accuracies and their standard deviation are given in Table II. WebTagger significantly exceeds the mean tagging accuracy compared to all state-of-the-art taggers. During the ten test runs we

TABLE II
Results for 10-fold cross validation trained on joint-domain data using *WebTrain*.

	WebTagger	TreeTagger	TnT	Stanford
Total	**94.09 ± 0.37**	93.72 ± 0.49	93.63 ± 0.37	93.18 ± 0.32
Known	**95.15 ± 0.37**	95.83 ± 0.43	95.81 ± 0.51	95.61 ± 0.40
Unknown	**72.75 ± 2.25**	67.98 ± 3.14	70.58 ± 2.08	68.14 ± 1.97
Percentage unknowns	**4.72 ± 0.40**	7.58 ± 0.75	8.65 ± 0.62	8.81 ± 0.62

perform 30 single comparisons with WebTagger. WebTagger performs in 28 of 30 cases better. Particularly, the accuracy on unknown tokens can be improved by our approach. Note that, the tagging accuracy for known tokens is slightly worse compared to state-of-the art taggers. But at the same time the number of unknown tokens is significantly reduced from 2.9 up to 4.1 percentage points compared to the state-of-the art taggers. Moreover, the accuracy for unknown tokens is increased. The main contribution for this is given by our approach combining prefix and suffix lexical probabilities by the arithmetic mean, which makes the method more robust for unknown tokens. Overall, considering the noisy characteristics of social media texts, a considerable enhancement is achieved with WebTagger. Precision and recall rates for each tag class are determined to investigate the tag/class-specific accuracies. Applying WebTagger leads to a mean precision of 0.86 with a standard deviation of 0.2. On average, a recall of 0.84 with standard deviation 0.23 is achieved. Considering the equally weighted combination of both measures, our approach results in a mean f_1-measure of 0.84.

The ten most frequently confused tag pairs for our approach are further investigated. The results are depicted in Table III.

TABLE III
Most frequently confused tag classes.

Correct	Predicted	Frequency
NE	**NN**	147
NN	**NE**	102
VVFIN	**VVINF**	90
KOM	ADV	58
FM	**NE**	58
NN	ADJA	46
PDS	ART	42
VVFIN	**VVPP**	40
VVINF	VVFIN	38
PRELS	ART	38

Bold classes also occur in the top ten confused tag classes, evaluated only for unknown tokens. Tagging errors are represented by absolute values/frequencies and calculated over all testing sets with approximately 36,000 tokens. The top two confusion pairs *noun* (NN) and *named entity* (NE) account for 12% of the errors. This is not a particular effect for social media texts, since it also occurs for newspaper texts. To distinguish proper nouns from named entities is done by named entity recognition and can not be solved by general POS taggers. Interchanging a *finite verb* (VVFIN) and a *non-finite verb* (VVINF) is caused by a non-local dependency particularly in German. This is also reported for state-of-the-art taggers and illustrated in [4]. Noticeable is the occurrence of tag confusion between *foreign language* (FM) and *named entity* (NE).

Social media texts are often multilingual and contain text parts written in different language, e.g., a German Web comment contains English text segments (FM). The tokens of such text segments are annotated as *foreign language* (FM). Due to the missing prefix/suffix information of such tokens, this leads to tagging errors. Frequent tag confusion between *noun* (NN) and *attributive adjective* (ADJA) results from missing noun capitalization, which causes a valid adjective, from self created tokens or token transformations.

Furthermore, we investigate the influence of training data selection and parameter estimation adaptions for lexical probabilities. Results are depicted in Table IV.

TABLE IV
Influence of training data and estimation methods

	Method	**Accuracy** (%)
Training	*TIGER* newspaper corpus	87.59 ± 1.21
	TIGER + *WebTrain* corpus, i.e., (+)	93.38 ± 0.42
Estimation	(+) + normalization ($t(w)$)	93.61 ± 0.44
	(+) + normalization ($t(w)$) + word classes ($\mathcal{W}_r$), i.e., ($\star$)	93.91 ± 0.39
	($\star$) + auxiliary lexicon ($\mathcal{L}^+$)	**94.09 ± 0.37**

A significant improvement is achieved by adding text genre specific training data, i.e., Web comments. We discuss this effect in detail in Section IV-C. The introduction of text normalization and regular expressions to build word classes leads to a significant improvement of 0.57 percentage points. Additional usage of an auxiliary lexicon, further increases accuracy about 0.18 percentage points.

All depicted results use a combined prefix/suffix lexicon to estimate lexical probabilities for (still) unknown tokens. Previous studies have shown, that adding prefix information for automatic tagging of newspaper texts only leads to little improvement of 0.05 percentage points, see [4]. In our approach running the tagger only with a suffix lexicon results in 0.3 and only with a prefix lexicon in 0.7 percentage points performance loss.

C. Influence of text genre-specific training data

To further investigate the influence of using text genre specific data, i.e., Web comments for training, we train our model based on different training corpora. First, we train our

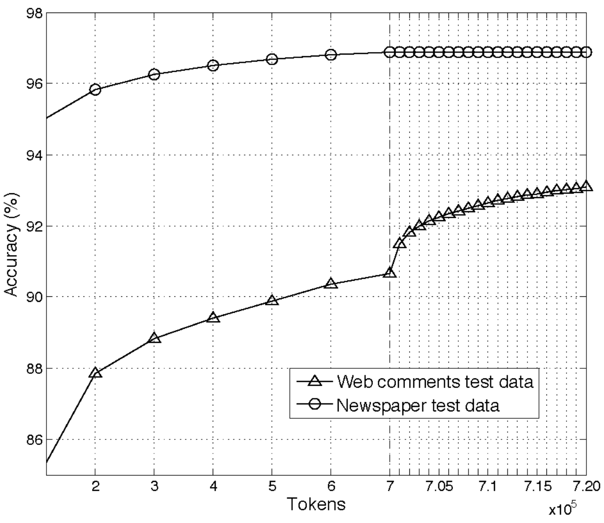

Fig. 1. Influence of additional newspaper/Web comment training, tested on Web comment and newspaper texts.

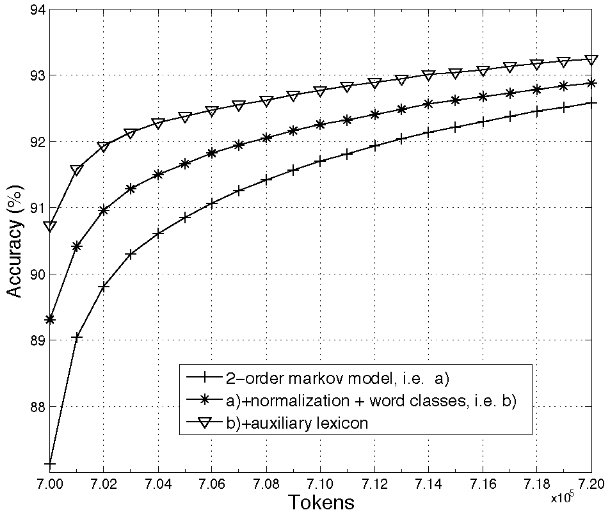

Fig. 2. Stepwise parameter estimation adaptions for increasing Web comment training data.

model exclusively on newspaper texts. We stepwise increase the amount of training data from 100,000 to 700,000 tokens. In each step we randomly choose sentences comprising 100,000 tokens. This is performed 100 times and data is added to the data selected in the previous step. Hence, the model is trained on 100 different samples in each step. Second, in twenty steps 1,000 up to 20,000 Web comment tokens are combined with a sample set of 700,000 newspaper training tokens. Here, we choose the newspaper training sample (700,000 tokens) achieving mean tagging accuracy, when tested on Web comments. Additional Web comment tokens are chosen randomly, sentence wise from *WebTrain*. Again we select 100 sample sets, in the same way as for the newspaper training and train our model on such data for each iteration step. Testing is performed on the remaining data, a fixed test set of Web comments with approximately 6,000 tokens. Mean results over 100 different trainings per point are depicted in the curve marked with $\triangle$ in Figure 1. The plot contains different x-axis scalings for the left and right area next to the black vertical line to better illustrate the results. Significant slope increase can be observed in this point, which proves the success by using text genre specific training data for the task of POS tagging for Web comments. Using 20,000 Web comment tokens results in approximately 2.4 percentage points performance improvement on average. Hence, little effort of manual annotation leads to a significant performance improvement. Increase of 600,000 newspaper training tokens results in approximately 5.8 percentage points improvement solely.

Furthermore, we show that including grammatically non-standardized texts as training data does not negatively effect the annotation of standardized text by means of the proposed approach. Random sentences are chosen from the newspaper *TIGER* corpus to create a test set of 90,000 newspaper tokens. We use WebTagger trained on the different training corpora to tag the newspaper data. The curve marked with $\circ$ in Figure 1 illustrates the results. Results proof that adding 20,000 Web comment tokens for training do not effect tagging accuracy for standardized texts essentially.

Comparison of tagging accuracies for Web comments and newspaper texts states that the tagging accuracy on standardized text can not be achieved when applying our approach to Web comments. However, the performance difference can be reduced from approximately 10 percentage points to 4 percentage points by increasing the amount of training data from 100,000 tokens to 720,000 tokens in total. Furthermore, matching the slope of both curves for the left area, states that increasing the amount of newspaper training data is more substantial for the application to Web comments compared to the application to newspaper texts. Tagging accuracy can be improved by 2 percentage points for newspaper data and 5.8 percentage points tested on Web comments by adding the same amount of newspaper training data.

Training our model on 700,000 *TIGER* tokens leads to similar results, when tested on newspaper data compared to TreeTagger results reported in [10]. For 90,000 randomly selected testing sentences chosen from the *TIGER* corpus, WebTagger achieves 96.9% accuracy on average.

Finally, the interaction between the amount of training data and the different adaption method for lexical probability estimation is illustrated in Figure 2. For testing the same 6,000 test tokens like in Figure 1 are used. We stepwise adapt the lexical parameter estimation method by our proposed methods, similarly to the procedure performed in Table IV. Significant impact of introducing text normalization and word classes is observed over the whole training data range. Using an auxiliary lexicon leads to a significant performance increase,

TABLE V
TAGGER EVALUATION FOR DIFFERENT TEXT TYPES TRAINED ON JOINT-DOMAIN DATA.

	#Tokens	WebTagger	TreeTagger	TnT	Stanford
WebTrain test	3,628	**94.09 ± 0.37**	93.72 ± 0.49	93.63 ± 0.37	93.18 ± 0.32
Chat messages	1,728	**91.75 ± 0.09**	89.12 ± 0.18	87.96 ± 0.11	87.81 ± 0.16
YouTube comments	1,463	**86.90 ± 0.19**	84.03 ± 0.24	81.18 ± 0.19	81.23 ± 0.16
Blog comments	815	**93.56 ± 0.29**	91.35 ± 0.18	90.46 ± 0.12	90.29 ± 0.17

particularly for a small amount of training data. Comparing the slopes of the curves marked with ∇ and ⋆ illustrates that the sufficient training data amount is much higher to compensate the improvement achieved by normalization and word classes methods. In total, all estimation adaptions can be partially compensated by adding additional Web comment training data at least for this test sample. This has to be studied in more detail for different test samples. However, manual annotation of complete texts for fully supervised training is a very time consuming step. Creating an auxiliary lexicon with our proposed method shows a better trade-off between time for annotation and improvement in tagging accuracy.

D. Transfer to other social media text types

In this subsection, we study the application of the proposed WebTagger to different social media text types, where the tagger is not trained on the particular type. To illustrate the improvements, Table V shows tagging accuracies and standard deviations for WebTagger and the three selected state-of-the art taggers. All taggers are trained on the joint-domain cross validation data described before. We compare the results for the particular Web comment test data to results achieved for blog comments, chat messages and YouTube comments from *WebTypes* corpus, introduced in Subsection IV-A.

Application of WebTagger leads to a consistent performance increase between approximately 2 and 6 percentage points for different social media text types. Best improvements can be observed for YouTube comments, which are highly characterized by a dialogue form and social media text characteristics, such as emoticons, word shortenings or letter iterations. Even though considerable improvement is achieved, the tagging accuracy of 86.9% is the lowest compared to all other types, due to the low text standardization. Overall, WebTagger outperforms the state-of-the art taggers for all social media text types.

Figure 3 shows the influence of additional Web comment training data to the different social media text types. The accuracy improvements over the different training data amounts are depicted in the corresponding curves. For all social media types the stepwise addition of *WebTrain* training data leads to a consistent accuracy increase. For *WebTypes* related text types, which show more social media text characteristics, the slope of the curves is higher compared to the particular training data type *WebTrain* (test, 6,000 tokens). Increasing the amount of Web comment training data leads to a significant performance increase, particularly for blog comments and YouTube comments. Results approve that general social media text characteristics can be learned from Web comments (*Heise*). In summary, the results from Table V and Figure 3 show that the adapted parameter estimation methods combined with a sufficient amount of Web comment training data leads to adequate tagging accuracies for social media texts in general. Results clearly demonstrate that the proposed tagger can successfully be applied to other texts belonging to the social media text genre.

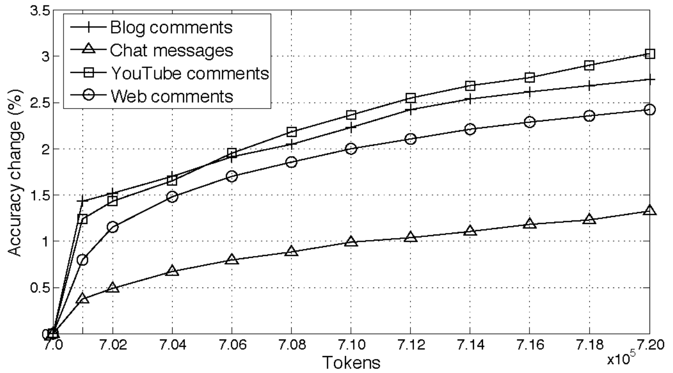

Fig. 3. Influence of additional Web comment training for different social media texts.

Note that we exclude twitter messages from this scope, since this subset can not be addressed suitably with the presently developed method. Due to their special characteristic given by hard distractions to 140 characters, the proposed method needs to be further adapted.

E. Transfer to other languages

The basic model and parameter estimation enhancements of the proposed WebTagger are language independent. It is adapted to the social media text characteristics in general, e.g., emoticons or character iterations. However, considering all minor effects that depend on language specific properties requires some additional effort, e.g., for adapting the normalization function. Moreover, language specific training would require an additional supervised social media text corpus. For the corpus annotation all described annotation rules can be used analogously. Evidently, POS tags need to be mapped to the language specific tag set.

V. Conclusion

A new POS tagger, WebTagger, designed for the annotation of social media texts has been presented. It yields a minimum improvement of 2.2 percentage points for different social media text types compared to state-of-the art taggers. Furthermore, WebTagger performs the best with an average accuracy of 94% evaluated in a cross validation on German Web comments. Our approach basically differs from other statistical Markov model taggers in estimation of lexical probabilities for unknown tokens. Before word classes realized by regular expressions and a prefix and suffix lexicon is adequately combined, a word preprocessing for text normalization is performed. Additionally, the usage of a semi-supervised auxiliary lexicon is proposed. Altogether, lexical probability distributions are estimated more accurately for social media texts.

Furthermore, the influence of manually annotated text genre specific training data, i.e., social media texts, is investigated. Considerable improvement is achieved by using only a small amount of 20,000 tokens as additional data for supervised training. Using such training data enables for reliable transition probability estimates by learning the different grammatical structure of social media texts.

In our approach we exemplarily use German social media texts. WebTaggers basic model and parameter estimation enhancements are language independent. However, we recommend a language specific training which requires an additional supervised social media text corpus.

Acknowledgment

This work was partially supported by the Project House HumTec at RWTH Aachen University, Germany. We would like to thank Phillip Vaßen for his contribution.

References

[1] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer, "Feature-rich Part-of-Speech Tagging With a Cyclic Dependency Network," in *Proceedings of Human Language Technology Conference*, 2003, pp. 173–180.

[2] P. Gadde, L. V. Subramaniam, and T. A. Faruquie, "Adapting a WSJ Trained Part-of-Speech Tagger to Noisy Text: Preliminary Results," in *Proceedings of the 2011 Joint Workshop on Multilingual OCR and Analytics for Noisy Unstructured Text Data*, 2011, pp. 5:1–5:8.

[3] H. Schmid, "Probabilistic Part-of-Speech Tagging Using Decision Trees," in *Proceedings of International Conference on New Methods in Language Processing*, 1994, pp. 44–49.

[4] ——, "Improvements in Part-of-Speech Tagging With an Application to German," in *Proceedings of the ACL SIGDAT-Workshop*, 1995, pp. 47–50.

[5] T. Brants, "TnT – A Statistical Part-of-Speech Tagger," in *Proceedings of the 6th Applied Natural Language Processing Conference*, 2000, pp. 224–231.

[6] A. Schiller, S. Teufel, C. Stöckert, and C. Thielen, "Guidelines für das Tagging deutscher Textcorpora mit STTS," 1999, university of Stuttgart.

[7] J. Giménez and L. Màrquez, "Svmtool: A General POS Tagger Generator Based on Support Vector Machines," in *Proceedings of the 4th International Conference on Language Resources and Evaluation*, 2004, pp. 43–46.

[8] H. Schmid, "Part-of-Speech Tagging With Neural Networks," in *Proceedings of the 15th Conference on Computational Linguistics*, 1994, pp. 172–176.

[9] M. Volk and G. Schneider, "Comparing a statistical and a rule-based tagger for German," in *Proceedings of the 4th Conference on Natural Language Processing*, 1998, pp. 125–137.

[10] E. Giesbrecht and S. Evert, "Is Part-of-Speech Tagging a Solved Task? An Evaluation of POS Taggers for the German Web as Corpus," in *Proceedings of the Fifth Web as Corpus Workshop*, 2009, pp. 27–35.

[11] A. Mikheev, "Automatic Rule Induction for Unknown Word Guessing," *Computational Linguistics*, vol. 23, pp. 405–423, 1997.

[12] H. Schütze, "Distributional Part-of-Speech Tagging," in *Proceedings of 7th Conference of the European Chapter of the Association for Computational Linguistics*, 1995, pp. 141–148.

[13] O. Owoputi, B. O'Connor, C. Dyer, K. Gimpel, and N. Schneider, "Part-of-Speech Tagging for Twitter: Word Clusters and Other Advances," School of Computer Science, Carnegie Mellon University, Tech. Rep., 2012.

[14] K. Gimpel, N. Schneider, B. O'Connor, D. Das, D. Mills, J. Eisenstein, M. Heilman, D. Yogatama, J. Flanigan, and N. A. Smith, "Part-of-Speech tagging for Twitter: annotation, features, and experiments," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, 2011, pp. 42–47.

[15] M. Neunerdt, M. Reyer, and R. Mathar, "Part-of-Speech Tagging for Social Media Texts," in *Proceedings of The International Conference of the German Society for Computational Linguistics and Language Technology*, 2013.

[16] B. Trevisan, M. Neunerdt, and E.-M. Jakobs, "A Multi-level Annotation Model for Fine-grained Opinion Detection in German Blog Comments," in *Proceedings of KONVENS 2012*, 2012, pp. 179–188.

[17] M. Beißwenger, M. Ermakova, A. Geyken, L. Lemnitzer, and A. Storrer, "A TEI Schema for the Representation of Computer-mediated Communication," *Journal of the Text Encoding Initiative*, no. 3, pp. 1–31, 2012. [Online]. Available: http://jtei.revues.org/476

[18] J. R. Quinlan, "Induction of Decision Trees," *Machine Learning*, pp. 81–106, 1986.

[19] S. Brants, S. Dipper, P. Eisenberg, S. Hansen-Schirra, E. König, W. Lezius, C. Rohrer, G. Smith, and H. Uszkoreit, "TIGER: Linguistic Interpretation of a German Corpus," *Research on Language & Computation*, pp. 597–620, 2004.

[20] M. Beißwenger, "Corpora zur computervermittelten (internetbasierten) Kommunikation," *Zeitschrift für germanistische Linguistik*, vol. 35, pp. 496–503, 2007.