A Robust Transformation-Based Learning Approach Using Ripple Down Rules for Part-of-Speech Tagging

Dat Quoc Nguyen [a,*,**], Dai Quoc Nguyen [b], Dang Duc Pham [c], and Son Bao Pham [d]
[a] *Department of Computing, Macquarie University, Australia*
E-mail: dat.nguyen@students.mq.edu.au
[b] *Department of Computational Linguistics, Saarland University, Germany*
E-mail: daiquocn@coli.uni-saarland.de
[c] *L3S Research Center, University of Hanover, Germany*
E-mail: pham@l3s.de
[d] *VNU University of Engineering and Technology, Vietnam National University, Hanoi, Vietnam*
E-mail: sonpb@vnu.edu.vn

Abstract. In this paper, we propose a new approach to construct a system of transformation rules for the Part-of-Speech (POS) tagging task. Our approach is based on an incremental knowledge acquisition method where rules are stored in an exception structure and new rules are only added to correct the errors of existing rules; thus allowing systematic control of the interaction between the rules. Experimental results on 13 languages show that our approach is fast in terms of training time and tagging speed. Furthermore, our approach obtains very competitive accuracy in comparison to state-of-the-art POS and morphological taggers.

Keywords: Natural language processing, Part-of-Speech tagging, Morphological tagging, Single Classification Ripple Down Rules, Rule-based POS tagger, RDRPOSTagger, Bulgarian, Czech, Dutch, English, French, German, Hindi, Italian, Portuguese, Spanish, Swedish, Thai, Vietnamese

1. Introduction

POS tagging is one of the most important tasks in Natural Language Processing (NLP) that assigns a tag to each word in a text, which the tag represents the word's lexical category [26]. After the text has been tagged or annotated, it can be used in many applications such as machine translation, information retrieval, information extraction and the like.

Recently, statistical and machine learning-based POS tagging methods have become the mainstream ones obtaining state-of-the-art performance. However, the learning process of many of them is time-consuming and requires powerful computers for training. For example, for the task of combined POS and morphological tagging, as reported by Mueller et al. [43], the taggers SVMTool [25] and CRFSuite [52] took 2,454 minutes (about 41 hours) and 9,274 minutes (about 155 hours) respectively to train on a corpus of 38,727 Czech sentences (652,544 words), using a machine with two Hexa-Core Intel Xeon X5680 CPUs with 3,33 GHz and 144 GB of memory. Therefore, such methods might not be reasonable for individuals having limited computing resources. In addition, the tagging speed of many of those systems is relatively slow. For example, as reported by Moore [42], the SVMTool, the COMPOST tagger [71] and the UPenn bidirectional tagger [66] respectively achieved the tagging speed of 7700, 2600 and 270 English word tokens per second, using a Linux workstation with Intel Xeon X5550 2.67 GHz processors. So these methods may not be adaptable to the recent large-scale data NLP tasks where the fast tagging speed is necessary.

Turning to the rule-based POS tagging methods, the most well-known method proposed by Brill [10] automatically learns transformation-based error-driven

[*]Corresponding author. E-mail: dat.nguyen@students.mq.edu.au.
[**]The first two authors contributed equally to this work.

rules. In the Brill's method, the learning process selects a new rule based on the temporary context which is generated by all the preceding rules; the learning process then applies the new rule to the temporary context to generate a new context. By repeating this process, a sequentially ordered list of rules is produced, where a rule is allowed to change the outputs of all the preceding rules, so a word could be relabeled multiple times. Consequently, the Brill's method is slow in terms of training and tagging processes [27, 46].

In this paper, we present a new error-driven approach to automatically restructure transformation rules in the form of a Single Classification Ripple Down Rules (SCRDR) tree [15, 57]. In the SCRDR tree, a new rule can only be added when the tree produces an incorrect output. Therefore, our approach allows the interaction between the rules, where a rule can only change the outputs of some preceding rules in a controlled context. To sum up, our contributions are:

– We propose a new transformation-based error-driven approach for POS and morphological tagging task, using SCRDR.[1] Our approach obtains fast performance in both learning and tagging process. For example, in the combined POS and morphological tagging task, our approach takes an average of 61 minutes (about 1 hour) to complete a 10-fold cross validation-based training on a corpus of 116K Czech sentences (about 1,957K words), using a computer with Intel Core i5-2400 3.1GHz CPU and 8GB of memory. In addition, in the English POS tagging, our approach achieves a tagging speed of 279K word tokens per second. So our approach can be used on computers with limited resources or can be adapted to the large-scale data NLP tasks.

– We provide empirical experiments on the POS tagging task and the combined POS and morphological tagging task for 13 languages. We compare our approach to two other approaches in terms of running time and accuracy, and show that our robust and language-independent method achieves a very competitive accuracy in comparison to the state-of-the-art results.

The paper is organized as follows: sections 2 and 3 present the SCRDR methodology and our new approach, respectively. Section 4 details the experimental results while Section 5 outlines the related work. Finally, Section 6 provides the concluding remarks and future work.

[1] Our free open-source implementation namely *RDRPOSTagger* is available at http://rdrpostagger.sourceforge.net/

2. SCRDR methodology

A SCRDR tree [15, 49, 57] is a binary tree with two distinct types of edges. These edges are typically called *except* and *if-not* edges. Associated with each node in the tree is a *rule*. A rule has the form: *if* α *then* β where α is called the *condition* and β is called the *conclusion*.

Cases in SCRDR are evaluated by passing a case to the root of the tree. At any node in the tree, if the condition of the rule at a node η is satisfied by the case (so the node η *fires*), the case is passed on to the *except* child node of the node η using the *except* edge if it exists. Otherwise, the case is passed on to the *if-not* child node of the node η. The conclusion of this process is given by the node which *fired* last.

For example, with the SCRDR tree in Figure 1, given a case of 5-word window context *"as/IN investors/NNS anticipate/VB a/DT recovery/NN"* where *"anticipate/VB"* is the current word and POS tag pair, the case satisfies the conditions of the rules at nodes (0), (1) and (4), then it is passed on to node (5), using *except* edges. As the case does not satisfy the condition of the rule at node (5), it is passed on to node (8) using the *if-not* edge. Also, the case does not satisfy the conditions of the rules at nodes (8) and (9). So we have the evaluation path (0)-(1)-(4)-(5)-(8)-(9) with the last fired node (4). Thus, the POS tag for *"anticipate"* is concluded as *"VBP"* produced by the rule at node (4).

A new node containing a new exception rule is added to an SCRDR tree when the evaluation process returns an *incorrect* conclusion. The new node is attached to the last node in the evaluation path of the given case with the *except* edge if the last node is the fired node; otherwise, it is attached with the *if-not* edge.

To ensure that a conclusion is always given, the root node (called the *default* node) typically contains a trivial condition which is always satisfied. The rule at the default node, the default rule, is the unique rule which is not an exception rule of any other rule.

In the SCRDR tree in Figure 1, rule (1) - the rule at node (1) - is an exception rule of the default rule (0). As node (2) is the *if-not* child node of node (1), rule (2) is also an exception rule of rule (0). Likewise, rule (3) is an exception rule of rule (0). Similarly, both rules (4) and (10) are exception rules of rule (1) whereas rules (5), (8) and (9) are exception rules of rule (4), and so on. Therefore, the exception structure of the SCRDR tree extends to four levels: rules (1), (2) and (3) at layer 1; rules (4), (10), (11), (12) and (14) at layer 2; rules (5), (8), (9), (13) and (15) at layer 3; and rules (6) and (7) at layer 4 of the exception structure.

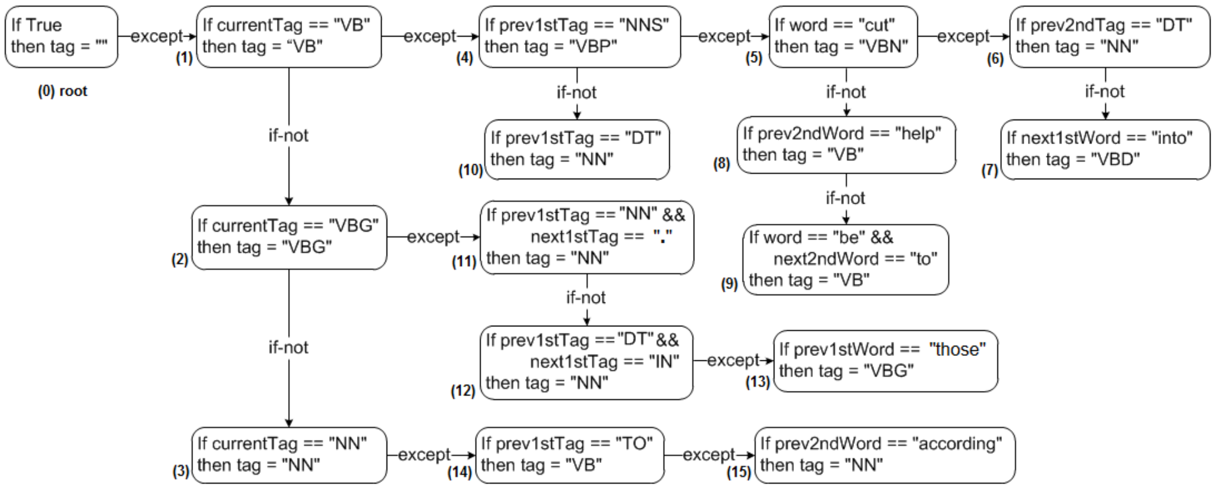
If True then tag = ""
(0) root
except
(1) If currentTag == "VB" then tag = "VB"
(4) If prev1stTag == "NNS" then tag = "VBP"
(5) If word == "cut" then tag = "VBN"
(6) If prev2ndTag == "DT" then tag = "NN"
if-not
(10) If prev1stTag == "DT" then tag = "NN"
(8) If prev2ndWord == "help" then tag = "VB"
(7) If next1stWord == "into" then tag = "VBD"
(9) If word == "be" && next2ndWord == "to" then tag = "VB"
(2) If currentTag == "VBG" then tag = "VBG"
(11) If prev1stTag == "NN" && next1stTag == "." then tag = "NN"
(12) If prev1stTag == "DT" && next1stTag == "IN" then tag = "NN"
(13) If prev1stWord == "those" then tag = "VBG"
(3) If currentTag == "NN" then tag = "NN"
(14) If prev1stTag == "TO" then tag = "VB"
(15) If prev2ndWord == "according" then tag = "NN"

Figure 1.: An example of a SCRDR tree for English POS tagging.

Template	Example
#2: *if* previous1st Word == **"object.previous1st Word"** *then* tag = **"correctTag"**	(13)
#3: *if* word == **"object.word"** *then* tag = **"correctTag"**	(5)
#4: *if* next1st Word == **"object.next1st Word"** *then* tag = **"correctTag"**	(7)
#10: *if* word == **"object.word"** && next2nd Word == **"object.next2nd Word"** *then* tag = **"correctTag"**	(9)
#15: *if* previous1st Tag == **"object.previous1st Tag"** *then* tag = **"correctTag"**	(4)
#20: *if* previous1st Tag == **"object.previous1st Tag"** && next1st Tag == **"object.next1st Tag"** *then* tag = **"correctTag"**	(11)

Table 1: Examples of rule templates corresponding to the rules (4), (5), (7), (9), (11) and (13) in Figure 1.

3. Our approach

In this section, we present a new error-driven approach to automatically construct a SCRDR tree of transformation rules for POS tagging. The learning process in our approach is described in Figure 2.

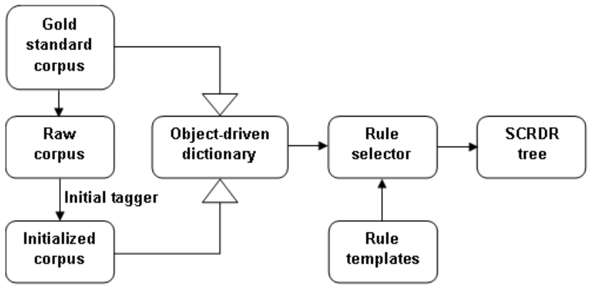

Figure 2.: The diagram of our learning process.

The *initialized corpus* is generated by using an *initial tagger* to perform POS tagging on the *raw corpus* which consists of the raw text extracted from the *gold standard training corpus*, excluding POS tags.

Our initial tagger uses a lexicon to assign a tag for each word. The lexicon is constructed from the gold standard corpus, where each word type is coupled with its most frequent associated tag in the gold standard corpus. In addition, the character 2-, 3-, 4- and 5-gram suffixes of word types are also included in the lexicon. Each suffix is coupled with the most frequent[2] tag associated to the word types containing this suffix. Furthermore, the lexicon also contains three default tags corresponding to the tags most frequently assigned to words containing numbers, capitalized words and lowercase words. The suffixes and default tags are only used to label unknown words (i.e. out-of-lexicon words).

To handle unknown words in English, our initial tagger uses regular expressions to capture the information about capitalization and word suffixes.[3] For other languages, the initial tagger firstly determines whether the word contains any numeric character to get the default tag for numeric word type. If the word does not contain any numeric character, the initial tagger then extracts

[2]The frequency must be greater than 1, 2, 3 and 4 for the 5-, 4-, 3- and 2-gram suffixes, respectively.

[3]An example of a regular expression in Python is as follows: *if (re.search(r'(.*ness$) | (.*ment$) | (.*ship$) | (^[Ee]x-.*) | (^[Ss]elf-.*)', word) != None): tag = "NN"*.

the 5-, 4-, 3- and 2-gram suffixes in this order and returns the coupled tag corresponding to the first suffix found in the lexicon. If the lexicon does not contain any of the suffixes of the word, the initial tagger determines whether the word is capitalized or in lowercase form to return the corresponding default tag.

By comparing the initialized corpus with the gold standard corpus, an *object-driven dictionary* of *Object* and *correctTag* pairs is produced. Each Object captures a 5-word window context of a word and its current initialized tag in the format of (*previous 2nd word, previous 2nd tag, previous 1st word, previous 1st tag, word, current tag, next 1st word, next 1st tag, next 2nd word, next 2nd tag, last-2-characters, last-3-characters, last-4-characters*), extracted from the initialized corpus.[4] The correctTag is the corresponding "true" tag of the word in the gold standard corpus.

words	w_{-2}, w_{-1}, w_0, w_{+1}, w_{+2}
word bigrams	(w_{-2}, w_0), (w_{-1}, w_0), (w_{-1}, w_{+1}), (w_0, w_{+1}) (w_0, w_{+2})
word trigrams	(w_{-2}, w_{-1}, w_0), (w_{-1}, w_0, w_{+1}), (w_0, w_{+1}, w_{+2})
POS tags	p_{-2}, p_{-1}, p_0, p_{+1}, p_{+2}
POS bigrams	(p_{-2}, p_{-1}), (p_{-1}, p_{+1}), (p_{+1}, p_{+2})
Combined	(p_{-1}, w_0), (w_0, p_{+1}), (p_{-1}, w_0, p_{+1}), (p_{-2}, p_{-1}, w_0) (w_0, p_{+1}, p_{+2})
suffixes	$c_{n-1}c_n$, $c_{n-2}c_{n-1}c_n$, $c_{n-3}c_{n-2}c_{n-1}c_n$

Table 2: Short descriptions of rule templates. "w" refers to word token and "p" refers to POS label while -2, -1, 0, 1, 2 refer to indices, for instance, p_0 indicates the current initialized tag. $c_{n-1}c_n$, $c_{n-2}c_{n-1}c_n$, $c_{n-3}c_{n-2}c_{n-1}c_n$ correspond to the character 2-, 3- and 4-gram suffixes of w_0. So the templates #2, #3, #4, #10, #15 and #20 in Table 1 are associated to w_{-1}, w_0, w_{+1}, (w_0, w_{+2}), p_{-1} and (p_{-1}, p_{+1}), respectively.

The *rule selector* is responsible for selecting the most suitable rules to build the *SCRDR tree*. To generate concrete rules, the rule selector uses *rule templates*. The examples of our rule templates are presented in Table 1, where the elements in **bold** will be replaced by specific values from the Object and correctTag pairs in the object-driven dictionary. Short descriptions of the rule templates are shown in Table 2.

The SCRDR rule tree is initialized with the default rule ***if*** *True* ***then*** *tag = ""* as shown in Figure 1.[5] Then the system creates a rule of the form ***if*** *currentTag ==* ***"Label"*** ***then*** *tag =* ***"Label"*** for each POS tag in the list of all tags extracted from the initialized corpus. These rules are added to the SCRDR tree as exception rules of the default rule to create the first layer exception structure, as for instance the rules (1), (2) and (3) in Figure 1.

3.1. Learning process

The process to construct new exception rules to higher layers of the exception structure in the SCRDR tree is as follows:

– At each node η in the SCRDR tree, let Θ_η be the set of Object and correctTag pairs from the object-driven dictionary such that the node η is the last fired node for every Object in Θ_η and the node η returns an incorrect POS tag (i.e. the POS tag concluded by the node η for each Object in Θ_η is not the corresponding correctTag). A new exception rule must be added to the next level of the SCRDR tree to correct the errors given by the node η.
– The new exception rule is selected from all concrete rules generated for all Objects in Θ_η. The selected rule must satisfy the following constraints: (i) If node η is at level-*k* exception structure in the SCRDR tree such that $k > 1$ then the rule's condition must not be satisfied by the Objects for which node η has already returned a correct POS tag. (ii) Let *A* and *B* be the number of Objects in Θ_η that satisfy the rule's condition, and the rule's conclusion returns the correct and incorrect POS tag, respectively. Then the rule with the highest score value $S = A - B$ will be chosen. (iii) The score *S* of the chosen rule must be higher than a given threshold. We apply two threshold parameters: the first threshold is to find exception rules at the layer-2 exception structure, such as rules (4), (10) and (11) in Figure 1, while the second threshold is to find rules for higher exception layers.
– If the learning process is unable to select a new exception rule, the learning process is repeated at node η_ρ for which the rule at the node η is an exception rule of the rule at the node η_ρ. Otherwise, the learning process is repeated at the new selected exception rule.

Illustration: To illustrate how new exception rules are added to build a SCRDR tree in Figure 1, we start with node (1) associated to rule (1) ***if*** *currentTag == "VB"* ***then*** *tag = "VB"* at the layer-1 exception structure. The learning process chooses the rule ***if*** *prev1stTag == "NNS"* ***then*** *tag = "VBP"* as an exception rule for rule (1). Thus, node (4) associated with

[4]In the example case from Section 2, the Object corresponding to the 5-word context window is {*as, IN, investors, NNS, anticipate, VB, a, DT, recovery, NN, te, ate, pate*}.

[5]The default rule returns an incorrect conclusion of empty POS tag for every Object.

rule (4) ***if*** *prev1st Tag == "NNS"* ***then*** *tag = "VBP"* is added as an *except* child node of node (1). The learning process is then repeated at node (4). Similarly, nodes (5) and (6) are added to the tree as shown in Figure 1.

The learning process now is repeated at node (6). At node (6), the learning process cannot find a suitable rule that satisfies the three constraints described above. So the learning process is repeated at node (5) because rule (6) is an exception rule of rule (5). At node (5), the learning process selects a new rule (7) ***if*** *next1st Word == "into"* ***then*** *tag = "VBD"* to be another exception rule of rule (5). Consequently, a new node (7) containing rule (7) is added to the tree as an *if-not* child node of node (6). At node (7), the learning process cannot find a new rule to be an exception rule of rule (7). Therefore, the learning process is again repeated at node (5).

This process of adding new exception rules is repeated until no rule satisfying the three constraints can be found.

3.2. Tagging process

The tagging process firstly tags unlabeled text by using the initial tagger. Next, for each initially tagged word the corresponding Object will be created by sliding a 5-word context window over the text from left to right. Finally, each word will be tagged by passing its Object through the learned SCRDR tree, as illustrated in the example in Section 2. If the default node is the last fired node satisfying the Object, the final tag returned is the tag produced by the initial tagger.

4. Empirical study

This section presents the experiments validating our proposed approach in 13 languages. We also compare our approach with the TnT[6] approach [9] and the MarMoT[7] approach proposed by Mueller et al. [43]. The TnT tagger is considered as one of the fastest POS taggers in literature (both in terms of training and tagging), obtaining competitive tagging accuracy on diverse languages [26]. The MarMoT tagger is a morphological tagger obtaining state-of-the-art tagging accuracy on various languages such as Arabic, Czech, English, German, Hungarian and Spanish.

We run all experiments on a computer of Intel Core i5-2400 3.1GHz CPU and 8GB of memory. Experiments on English use the Penn WSJ Treebank [40] sections 0-18 (38,219 sentences - 912,344 words) for training, sections 19-21 (5,527 sentences - 131,768 words) for validation, and the sections 22-24 (5,462 sentences - 129,654 words) for testing. The proportion of unknown words in the test set is 2.81% (3,649 unknown words). We also conduct experiments on 12 other languages. The experimental datasets for those languages are described in Table 3.

Apart from English, it is difficult to compare the results of previously published works because each of them have used different experimental setups and data splits. Thus, it is difficult to create the same evaluation settings used in the previous works. So we perform 10-fold cross validation[8] for all languages other than English, except for Vietnamese where we use 5-fold cross validation.

Our approach: In training phase, all words appearing only once time in the training set are initially treated as unknown words and tagged as described in Section 3. This strategy produces tagging models containing transformation rules learned on error contexts of unknown words. The threshold parameters were tuned on the English validation set. The best value pair (3, 2) was then used in all experiments for all languages.

TnT & **MarMoT**: We used default parameters for training TnT and MarMoT.

4.1. Accuracy Results

We present the tagging accuracy of our approach with the lexicon-based initial tagger (for short, RDRPOSTagger) and TnT in Table 4. As can be seen from Table 4, our RDRPOSTagger does better than TnT on isolating languages such as Hindi, Thai and Vietnamese. For the combined POS and morphological (POS+MORPH) tagging task on morphologically rich languages such as Bulgarian, Czech, Dutch, French, German, Portuguese, Spanish and Swedish, RDRPOSTagger and TnT generally obtain similar results on known words. However, RDRPOSTagger performs worse on unknown words. This can be because RDRPOSTagger uses a simple lexicon-based method for tagging unknown words, while TnT uses a more complex suffix analysis to handle unknown words.

[6]www.coli.uni-saarland.de/~thorsten/tnt/
[7]http://cistern.cis.lmu.de/marmot/

[8]For each dataset, we split the dataset into 10 contiguous parts (i.e. 10 contiguous folds). The evaluation procedure is repeated 10 times. Each part is used as the test set and 9 remaining parts are merged as the training set. *All accuracy results are reported as the average results over the test folds.*

Language	Source	#sen	#words	#P	#PM	OOV
Bulgarian	BulTreeBank-Morph [67]	20,558	321,538	—	564	10.07
Czech	PDT Treebank 2.5 [5]	115,844	1,957,246	—	1,570	6.09
Dutch	Lassy Small Corpus [51]	65,200	1,096,177	—	933	7.21
French	French Treebank [1]	21,562	587,687	17	306	5.19
German	TIGER Corpus [8]	50,474	888,236	54	795	7.74
Hindi	Hindi Treebank [55]	26,547	588,995	39	—	—
Italian	ISDT Treebank [7]	10,206	190,310	70	—	11.57
Portuguese	Tycho Brahe Corpus [21]	68,859	1,482,872	—	344	4.39
Spanish	IULA LSP Treebank [41]	42,099	589,542	—	241	4.94
Swedish	Stockholm—Umeå Corpus 3.0 [72]	74,245	1,166,593	—	153	8.76
Thai	ORCHID Corpus [70]	23,225	344,038	47	—	5.75
Vietnamese	(VTB) Vietnamese Treebank [50]	10,293	220,574	22	—	3.41
	(VLSP) VLSP Evaluation Campaign 2013	28,232	631,783	31	—	2.06

Table 3: The experimental datasets. **#sen**: the number of sentences. **#words**: the number of words. **#P**: the number of POS tags. **#PM**: the number of combined POS and morphological (POS+MORPH) tags. **OOV** (Out-of-Vocabulary): the average percentage of unknown word tokens in each test fold. For Hindi, OOV rate is 0.0% on 9 test folds while it is 3.8% on the remaining test fold.

Language	RDRPOSTagger								TnT				
	Initial accuracy			Tagging accuracy			Speed		Tagging accuracy			Speed	
	Kno.	Unk.	All.	Kno.	Unk.	All.	TT	TS	Kno.	Unk.	All.	TT	TS
Bulgarian*	95.13	49.50	90.53	96.59	66.06	93.50	2	157K	96.55	70.10	**93.86**$^+$	1	313K
Czech*	84.05	52.60	82.13	93.01	64.86	91.29	61	56K	92.95	67.83	**91.42**$^+$	1	164K
Dutch*	88.91	54.30	86.34	93.88	60.15	91.39	44	103K	93.32	69.07	**91.53**	1	125K
English	93.94	78.84	93.51	96.91	83.89	**96.54**$^+$	18	279K	96.77	86.02	96.46	1	720K
French	95.99	77.18	94.99	98.07	81.57	**97.19**	16	237K	97.52	87.43	96.99	1	722K
French*	89.97	54.36	88.12	95.09	63.74	93.47	9	240K	95.13	70.67	**93.88**$^+$	1	349K
German	94.76	73.21	93.08	97.74	78.87	96.28	28	212K	97.70	89.38	**97.05**$^+$	1	509K
German*	71.68	30.92	68.52	87.70	51.84	84.92	22	111K	86.98	61.22	**84.97**	1	98K
Hindi	__	__	89.63	__	__	**95.77**$^+$	21	210K	__	__	94.80	1	735K
Italian	92.63	67.33	89.59	95.93	71.79	93.04	3	276K	96.38	86.16	**95.16**$^+$	1	446K
Portuguese*	92.85	61.19	91.43	96.07	64.38	94.66	42	172K	96.01	78.81	**95.24**$^+$	1	280K
Spanish*	97.94	75.63	96.92	98.85	79.50	97.95	4	283K	98.96	84.16	**98.18**	1	605K
Swedish*	90.85	71.60	89.19	96.41	76.04	94.64	41	152K	96.33	85.64	**95.39**$^+$	1	326K
Thai	92.17	75.91	91.23	94.98	80.68	**94.15**$^+$	6	315K	94.32	80.93	93.54	1	490K
Vn (VTB)	92.17	55.21	90.90	94.10	56.38	**92.80**$^+$	5	269K	92.90	59.35	91.75	1	723K
(VLSP)	91.88	64.36	91.31	94.12	65.38	**93.53**$^+$	23	145K	92.65	68.07	92.15	1	701K

Table 4: The accuracy results (%) of our approach using the lexicon-based initial tagger (for short, RDRPOSTagger) and TnT. Languages marked with * indicate the tagging accuracy on combined POS+MORPH tags. **Vn** $\rightarrow$ Vietnamese. **Kno.**: the known word tagging accuracy. **Unk.**: the unknown word tagging accuracy. **All.**: the overall accuracy result. **TT**: training time (minutes). **TS**: tagging speed (number of word tokens per second). Higher results are highlighted in **bold**. Results marked $^+$ refer to a significant test with p-value < 0.05, using the two sample Wilcoxon test; due to a non-cross validation evaluation, we used accuracies over POS labels to perform significance test for English.

Therefore, TnT performs better than RDRPOSTagger on morphologically rich languages.

These initial accuracy results could be improved by following any of the previous studies that use external lexicon resources or existing morphological analyzers. In this research work, we simply employ TnT as the initial tagger in our approach. We report the accuracy results of our approach using TnT as the initial tagger (for short, RDRPOSTagger$_{\text{+TnT}}$) and MarMoT in Table 5. To sum up, RDRPOSTagger$_{\text{+TnT}}$ obtains competitive results in comparison to the state-of-the-art MarMoT tagger, across the 13 experimental

Language	**RDRPOSTagger$_{+TnT}$**			**MarMoT**				
	Tagging accuracy			Tagging accuracy			Speed	
	Kno.	Unk.	All.	Kno.	Unk.	All.	TT	TS
Bulgarian*	96.82	70.27	94.12	96.92	76.72	**94.86**$^+$	9	4K
Czech*	93.24	67.92	91.70	94.74	75.84	**93.59**$^+$	130	2K
Dutch*	94.00	69.20	92.17	94.74	73.39	**93.17**$^+$	44	3K
English	97.17	86.19	96.86	97.47	89.39	**97.24**	5	16K
French	98.27	87.55	97.70	98.33	91.15	**97.93**	2	12K
French*	95.42	70.93	94.16	95.55	77.66	**94.62**$^+$	9	6K
German	98.13	89.43	97.46	98.30	92.54	**97.85**$^+$	5	9K
German*	87.65	62.05	85.66	90.61	69.13	**88.94**$^+$	32	3K
Hindi	__	__	96.21	__	__	**96.61**$^+$	3	16K
Italian	96.75	86.18	95.49	96.90	89.21	**95.98**$^+$	2	6K
Portuguese*	96.30	78.81	95.53	96.53	81.49	**95.86**$^+$	23	6K
Spanish*	99.05	84.13	98.26	99.08	86.86	**98.45**$^+$	8	8K
Swedish*	96.79	85.68	95.81	97.15	86.63	**96.22**$^+$	11	7K
Thai	95.03	81.10	94.21	95.42	86.99	**94.94**$^+$	2	12K
Vn (VTB)	94.15	59.39	92.95	94.37	69.89	**93.53**$^+$	1	16K
(VLSP)	94.16	68.14	93.63	94.52	75.36	**94.13**$^+$	3	21K

Table 5: The accuracy results (%) of our approach using TnT as the initial tagger (for short, RDRPOSTagger$_{+TnT}$) and MarMoT.

languages. In particular, excluding Czech and German where MarMoT embeds existing morphological analyzers, RDRPOSTagger$_{+TnT}$ obtains accuracy results which mostly are about 0.5% lower than MarMoT's.

4.1.1. English

RDRPOSTagger produces a SCRDR tree model of 2,549 rules in a 5-level exception structure and achieves an accuracy of 96.54% against 96.46% accounted for TnT, as presented in Table 4. Table 6 presents the accuracy results obtained up to each exception level of the tree.

Level	Number of rules	Accuracy
$<= 1$	47	93.51 %
$<= 2$	1,522	96.36 %
$<= 3$	2,503	96.53 %
$<= 4$	2,547	96.54 %
$<= 5$	2,549	96.54 %

Table 6: Results due to levels of exception structures.

As shown in [47], using the same evaluation scheme for English, the Brill's rule-based tagger V1.14 [10] gained a similar accuracy result at 96.53%.[9] Using TnT as the initial tagger, RDRPOSTagger$_{+TnT}$ achieves an accuracy of 96.86% which is comparable to the state-of-the-art result at 97.24% obtained by MarMoT.

[9]The Brill's tagger uses an initial tagger with an accuracy of 93.58% on the test set. Using this initial tagger, our approach gains a higher accuracy of 96.57%.

4.1.2. Bulgarian

In Bulgarian, RDRPOSTagger$_{+TnT}$ obtains an accuracy of 94.12% which is 0.74% lower than the accuracy of MarMoT at 94.86%.

This is better than the results reported on the BulTreeBank webpage[10] on POS+MORPH tagging task, where TnT, SVMTool [25] and the memory-based tagger in the Acopost package[11] [64] obtained accuracies of 92.53%, 92.22% and 89.91%, respectively. Our result is also better than the accuracy of 90.34% reported by Georgiev et al. [22], obtained with the Maximum Entropy-base POS tagger from the OpenNLP toolkit.[12] Recently, Georgiev et al. [23][13] reached the state-of-the-art accuracy result of 97.98% for POS+MORPH tagging, however, without external resources the accuracy was 95.72%.

4.1.3. Czech

Mueller et al. [43] presented the results of five POS taggers SVMTool, CRFSuite [52], RFTagger [62], Morfette [12] and MarMoT for Czech POS+MORPH tagging. All models were trained using a training set of 38,727 sentences (652,544 tokens) and evaluated on a test set of 4,213 sentences (70,348 tokens), extracted

[10]http://www.bultreebank.org/taggers/taggers.html

[11]http://acopost.sourceforge.net/

[12]http://opennlp.sourceforge.net

[13]Georgiev et al. [23] split the BulTreeBank corpus into training set of 16,532 sentences, development set of 2,007 sentences and test set of 2,017 sentences.

from the Prague Dependency Treebank 2.0. The accuracy results are 89.62%, 90.97%, 90.43%, 90.01% and 92.99% accounted for SVMTool, CRFSuite, RFTagger, Morfette and MarMoT, respectively.

Since we could not access the Czech datasets used in the experiments above, we employ the Prague Dependency Treebank 2.5 [5] containing about 116K sentences. The accuracies of RDRPOSTagger (91.29%) and RDRPOSTagger$_{+\text{TnT}}$ (91.70%) compare favorably to the result of MarMot (93.50%).

4.1.4. Dutch

The TADPOLE tagger [78] was reached an accuracy of 96.5% when trained on a manually POS-annotated corpus containing 11 million Dutch words and 316 tags. Due to the limited access we could not use this corpus in our experiments and thus we can not compare our results with the TADPOLE tagger. Instead, we use the Lassy Small Corpus [51] containing about 1.1 million words. RDRPOSTagger$_{+\text{TnT}}$ achieves a promising accuracy at 92.17% which is 1% absolute lower than the accuracy of MarMoT (93.17%).

4.1.5. French

Current state-of-the-art methods for French POS tagging have reached accuracies up to 97.75% [65, 17], using the French Treebank [1] with 9,881 sentences for training and 1,235 sentences for test. However, these methods employed Lefff [58] which is an external large-scale morphological lexicon. Without using the lexicon, Denis and Sagot [17] reported an accuracy performance at 97.0%.

We trained our systems on 21,562 annotated French Treebank sentences and gained a POS tagging accuracy of 97.70% using RDRPOSTagger$_{+\text{TnT}}$ model, which is comparable to the accuracy at 97.93% of MarMoT. Regarding to POS+MORPH tagging, as far as we know this is the first experiment for French, where RDRPOSTagger$_{+\text{TnT}}$ obtains an accuracy of 94.16% against 94.62% obtained by MarMoT.

4.1.6. German

Using the 10-fold cross validation evaluation scheme on the TIGER corpus [8] of 50,474 German sentences, Giesbrecht and Evert [24] presented the results of TreeTagger [61], TnT, SVMTool, Stanford tagger [75] and Apache UIMA Tagger[14] obtaining the POS tagging accuracies at 96.89%, 96.92%, 97.12%, 97.63% and 96.04%, respectively. In the same evaluation setting, RDRPOSTagger$_{+\text{TnT}}$ gains an accuracy result of 97.46% while MarMoT gains a higher accuracy at 97.85%.

Turning to POS+MORPH tagging, Mueller et al. [43] also performed experiments on the TIGER corpus, using 40,474 sentences for training and 5,000 sentences for test. They presented accuracy performances of 83.42%, 85.68%, 84.28%, 83.48% and 88.58% obtained with the taggers SVMTool, CRFSuite, RFTagger, Morfette and MarMoT, respectively. In our evaluation scheme, RDRPOSTagger and RDRPOSTagger$_{+\text{TnT}}$ correspondingly achieve favorable accuracy results at 84.92% and 85.66% in comparison to an accuracy at 88.94% of MarMoT.

4.1.7. Hindi

On the Hindi Treebank [55], RDRPOSTagger$_{+\text{TnT}}$ reaches a competitive accuracy result of 96.21% against the accuracy of MarMoT at 96.61%. Being one of the largest languages in the world, there are many previous works on POS tagging for Hindi. However, most of them have used small manually labeled datasets that are not publicly available and that are smaller than the Hindi Treebank used in this paper.

Joshi et al. [29] achieved an accuracy of 92.13% using a Hidden Markov Model-based approach, trained on a dataset of 358K words and tested on 12K words. Using another training set of 150K words and test set of 40K words, Agarwal et al. [2] compared machine learning-based approaches and presented the POS tagging accuracy at 93.70%.

In the 2007 Shallow Parsing Contest for South Asian Languages [6], the POS tagging track provided a small training set of 21,470 words and a test set of 4,924 words. The highest accuracy in the contest was 78.66% obtained by Avinesh and Karthik [4]. In the same 4-fold cross validation evaluation scheme using a dataset of 15,562 words, Singh et al. [68] obtained an accuracy of 93.45% whilst Dalal et al. [16] achieved a result at 94.38%.

4.1.8. Italian

In the EVALITA 2009 workshop on Evaluation of NLP and Speech Tools for Italian[15], the POS tagging track [3] provided a training set of 3,719 sentences (108,874 word forms) with 37 POS tags. The teams participating in the closed task where using external resources was not allowed achieved various tagging accuracies on a test set of 147 sentences (5,066 word forms), ranging from 93.21% to 96.91%.

Our experiment on Italian POS tagging employs the ISDT Treebank [7] of 10,206 sentences (190,310 word

[14]https://uima.apache.org/sandbox.html#tagger.annotator

[15]http://www.evalita.it/2009

forms) with 70 POS tags. RDRPOSTagger$_{\text{+TnT}}$ obtains a competitive accuracy performance at 95.49% against 95.98% computed for MarMoT.

4.1.9. Portuguese

The previous works [18, 30] on POS+MORPH tagging for Portuguese used an early version of the Tycho Brahe corpus [21] containing about 1,036K words. The corpus was split into a training set of 776K words and a test set of 260K words. Based on this setting, Kepler and Finger [30] achieved an accuracy of 95.51% while dos Santos et al. [18] reached a state-of-the-art accuracy result at 96.64%.

The Tycho Brahe corpus in our experiment consists of about 1,639K words. RDRPOSTagger$_{\text{+TnT}}$ reaches an accuracy at 95.53% while MarMoT obtains higher result at 95.86% on 10-fold cross validation.

4.1.10. Spanish

In addition to Czech and German, Mueller et al. [43] evaluated the five taggers of SVMTool, CRFSuite, RFTagger, Morfette and MarMoT for Spanish POS+MORPH tagging, using a training set of 14,329 sentences (427,442 tokens) and a test set of 1,725 sentences (50,630 tokens) with 303 POS+MORPH tags. The accuracy results of the five taggers ranged from 97.35% to 97.93%, in which MarMoT obtained the highest result.

As we could not access the training and test sets used in Mueller et al. [43]'s experiment, we use the IULA Spanish LSP Treebank [41] of 42K sentences with 241 tags. RDRPOSTagger and RDRPOSTagger$_{\text{+TnT}}$ achieve accuracies of 97.95% and 98.26%, respectively, while MarMoT obtains a higher result at 98.45%.

NOTE that here we can make an indirect comparison between our RDRPOSTagger and the SVMTool, CRFSuite, RFTagger and Morfette taggers via MarMoT. We conclude that the results of RDRPOSTagger would likely be similar to the results of SVMTool, CRFSuite, RFTagger and Morfette on Spanish as well as on Czech and German.

4.1.11. Swedish

On the same SUC corpus 3.0 [72] consisting of 500 text files with about 74K sentences that we also use, Östling [53] evaluated the Swedish POS tagger Stagger using 10-fold cross validation but the folds were split at the file level and not on sentence level as we do. Stagger attained an accuracy of 96.06%.

In our experiment, RDRPOSTagger$_{\text{+TnT}}$ obtains an accuracy result of 95.81% in comparison to the accuracy at 96.22% of MarMoT.

4.1.12. Thai

On the Thai POS Tagged corpus ORCHID [70] of 23,225 sentences, RDRPOSTagger$_{\text{+TnT}}$ achieves an accuracy of 94.22% which is 0.72% absolute lower than the accuracy result of MarMoT (94.94%).

It is difficult to compare our results to the previous work on Thai POS tagging. For example, the previous works [39, 45] performed their experiments on an unavailable corpus of 10,452 sentences. The ORCHID corpus was also used in a POS tagging experiment presented by Kruengkrai et al. [32], however, the obtained accuracy of 79.342% was dependent on the performance of automatic word segmentation. On another corpus of 100K words, Pailai et al. [54] reached an accuracy of 93.64% using 10-fold cross validation.

4.1.13. Vietnamese

We participated in the first evaluation campaign on Vietnamese language processing[16] (VLSP). The campaign's POS tagging track provided a training set of 28,232 POS-annotated sentences and an unlabeled test set of 2,130 sentences. RDRPOSTagger achieved the 1st place in the POS tagging track.

In this paper, we also carry out POS tagging experiments using 5-fold cross validation evaluation scheme on the VLSP set of 28,232 sentences and the standard benchmark Vietnamese Treebank [50] of about 10K sentences. On these datasets, RDRPOSTagger$_{\text{+TnT}}$ achieves competitive results (93.63% and 92.95%) in comparison to MarMoT (94.13% and 93.53%).

In addition, on the Vietnamese Treebank, RDRPOSTagger with the accuracy 92.59% outperforms the previously reported Maximum Entropy Model, Conditional Random Fields and Support Vector Machine-based approaches [76] where the highest obtained accuracy was 91.64%.

4.2. Training time and tagging speed

While most published works have not reported training times and tagging speeds, we present our single-threaded implementation results in Tables 4 and 5.[17] From there we can see that TnT is the fastest in terms of both training and tagging when compared to our RDRPOSTagger and MarMoT. Our RDRPOSTagger and MarMoT require similar training times, however, RDRPOSTagger is significantly faster than MarMoT in terms of tagging speed.

[16]http://uet.vnu.edu.vn/rivf2013/campaign.html

[17]To measure the tagging speed on a test fold, we perform the tagging process on the test fold 10 times and then take the average.

It is interesting to note that in some languages, training our RDRPOSTagger is faster for combined POS+MORPH tagging task than for POS tagging, as presented in experimental results for French (9 minutes vs 16 minutes) and German (22 minutes vs 28 minutes) in Table 4. Usually in machine learning-based approaches fewer number of tags leads to higher training speed. For example, on a 40,474-sentence subset of the German TIGER corpus [8], SVMTool took about 899 minutes (about 15 hours) to train using 54 POS tags as compared to about 1,649 minutes (about 27 hours) using 681 POS+MORPH tags [43].

Language	**#sent**	**#tags**	SVMT	Morf	CRFS	RFT
German	40,474	681	1,649	286	1,295	5
Czech	38,727	1,811	2,454	539	9,274	3
Spanish	14,329	303	64	63	69	1

Table 7: The training time in minutes reported by Mueller et al. [43] for POS+MORPH tagging on a machine of two Hexa-Core Intel Xeon X5680 CPUs with 3,33 GHz and 144 GB of memory. **#sent**: the number of sentences in training set. **#tag**: the number of POS+MORPH tags. SVMT: SVMTool, Morf: Morfette, CRFS: CRFSuite, RFT: RFTagger.

In order to compare with other existing POS taggers in terms of the training time, we show in Table 7 the time taken to train the SVMTool, CRFSuite, Morfette and RFTagger using a more powerful computer than ours. For instance, on the German TIGER corpus, RDRPOSTagger took an average of 22 minutes to train a POS+MORPH tagging model while SVMTool and CRFSuite took 1,649 minutes (about 27 hours) and 1,295 minutes (about 22 hours) respectively, as shown in Table 7. Furthermore, RDRPOSTagger uses larger datasets for Czech and Spanish and obtains faster training process as compared to SVMTool, CRFSuite and Morfette.

Regarding to tagging speed, as reported by Moore [42] using the same evaluation scheme on English on a Linux workstation equipped with Intel Xeon X5550 2.67 GHz: the SVMTool, the UPenn bidirectional tagger [66], the COMPOST tagger [71], Moore [42]'s approach, the accurate version of the Stanford tagger [75] and the fast and less accurate version of the Stanford tagger gained tagging speed of 7700, 270, 2600, 51K, 5900 and 80K tokens per second, respectively. In our experiment, RDRPOSTagger obtains a faster tagging speed of 279K tokens per second on a weaker computer. To the best of our knowledge, we conclude that RDRPOSTagger is fast both in terms of training and tagging in comparison to other approaches.

5. Related work

From early POS tagging approaches the rule-based Brill's tagger [10] is the most well-known. The key idea of the Brill's method is to compare a manually annotated gold standard corpus with an initialized corpus which is generated by executing an initial tagger on the corresponding unannotated corpus. Based on the predefined rule templates, the method then automatically produces a list of concrete rules to correct wrongly assigned POS tags. For example, the template *"transfer tag of current word from **A** to **B** if the next word is **W**"* can produce concrete rules such as *"transfer tag of current word from **JJ** to **NN** if the next word is **of**"* or *"transfer tag of current word from **VBD** to **VBN** if the next word is **by**."*

At each training iteration, the Brill's tagger generates a set of all possible rules and chooses the ones that help to correct the incorrectly tagged words in the whole corpus. Thus, the Brill's training process takes a significant amount of time. To prevent that, Hepple [27] presented an approach with two assumptions for disabling interactions between rules to reduce the training time while sacrificing a small amount of accuracy. Ngai and Florian [46] proposed another method to reduce the training time by recalculating the scores of rules while obtaining similar accuracy result.

The *main difference* between our approach and the Brill's method is that we construct transformation rules in the form of a SCRDR tree where a new transformation rule is produced only based on a subset of tagging errors. So our approach is faster in term of training speed. In the conference publication version of our approach [47], we reported an improvement up to 33 times in training speed against the Brill's method. In addition, the Brill's method enables each subsequent rule to change the outputs of all preceding rules, thus a word can be tagged multiple times in the tagging process, each time by a different rule. This is different from our approach where each word is tagged only once. Consequently, our approach also achieves a faster tagging speed.

In addition to our research, there is only one work that applies Ripple Down Rules method for POS tagging proposed by Xu and Hoffmann [79]. Though Xu and Hoffmann's method obtained a very competitive accuracy, it is a hand-crafted approach taking about 60 hours to manually build a SCRDR tree model for English POS tagging.

Turning to statistical and machine learning methods for POS tagging, these methods can be listed as various Hidden Markov model-based methods [9, 20, 73], maximum entropy-based methods [12, 56, 74, 75, 77], perceptron algorithm-based approaches [13, 66, 71], neural network-based approaches [11, 14, 33, 38, 59, 60, 80], Conditional Random Fields [34, 35, 37, 43, 44], Support Vector Machines [25, 31, 63, 69] and other approaches including decision trees [61, 62] and hybrid methods [19, 36]. Overview about the POS tagging task can be found in [26, 28].

6. Conclusion and future work

In this paper, we propose a new error-driven method to automatically construct a Single Classification Ripple Down Rules tree of transformation rules for POS and morphological tagging. Our method allows the interaction between rules where a rule only changes the results of a limited number of other rules. Experimental evaluations for POS tagging and the combined POS and morphological tagging on 13 languages show that our method obtains very promising accuracy results. In addition, we successfully achieve fast training and tagging processes for all experimental languages. This could help to significantly reduce time and effort for the machine learning tasks on big data, employing POS and morphological information as learning features.

An important point is that our approach is suitable to involve domain experts to add new exception rules given concrete cases that are misclassified by the tree model. This is especially important for under-resourced languages where obtaining a large annotated corpus is difficult. In future work, we plan to build tagging models for other languages such as Russian, Arabic, Latin, Hungarian, Chinese and so forth.

Bibliographic note

This paper extends the work published in our conference publications [47, 48]. We make minor revisions to our published approach to yield improved accuracy results on English and Vietnamese, and we conduct new extensive empirical study on 11 other languages.

Acknowledgments

This research work is partially supported by the research project "VNU-SMM: An automated system for monitoring online social media to assist management and support decision making in business, politics, education, and social areas" from Vietnam National University, Hanoi. The first author is supported by an International Postgraduate Research Scholarship and a NICTA NRPA Top-Up Scholarship. The authors would like to thank the three anonymous reviewers, the associate editor Prof. Fabrizio Sebastiani and Dr. Kairit Sirts at the Macquarie University, Australia for helpful comments and suggestions.

References

[1] A. Abeillé, L. Clément, and F. Toussenel. Building a Treebank for French. In *Treebanks*, volume 20 of *Text, Speech and Language Technology*, pages 165–187. 2003.

[2] M. Agarwal, R. Goutam, A. Jain, S. R. Kesidi, P. Kosaraju, S. Muktyar, B. Ambati, and R. Sangal. Comparative Analysis of the Performance of CRF, HMM and MaxEnt for Part-of-Speech Tagging, Chunking and Named Entity Recognition for a Morphologically. In *Proceedings of the 12th Conference of the Pacific Association for Computational Linguistics*, pages 3–6, 2011.

[3] G. Attardi and M. Simi. Overview of the EVALITA 2009 Part-of-Speech Tagging Task. In *Poster and Workshop Proceedings of the 11th Conference of the Italian Association for Artificial Intelligence*, 2009.

[4] P. Avinesh and G. Karthik. Part-Of-Speech Tagging and Chunking using Conditional Random Fields and Transformation Based Learning. In *Proceedings of IJCAI 2007 Workshop on Shallow Parsing for South Asian Languages*, 2007.

[5] E. Bejcek, J. Panevová, J. Popelka, P. Stranák, M. Sevcíková, J. Stepánek, and Z. Zabokrtský. Prague Dependency Treebank 2.5 - a Revisited Version of PDT 2.0. In *Proceedings of 24th International Conference on Computational Linguistics*, pages 231–246, 2012.

[6] A. Bharathi and P. R. Mannem. Introduction to the Shallow Parsing Contest for South Asian Languages. In *Proceedings of IJCAI 2007 Workshop on Shallow Parsing for South Asian Languages*, 2007.

[7] C. Bosco, S. Montemagni, and M. Simi. Converting Italian Treebanks : Towards an Italian Stanford Dependency Treebank. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 61–69, 2013.

[8] 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 and Computation*, 2(4):597–620, 2004.

[9] T. Brants. TnT: A Statistical Part-of-Speech Tagger. In *Proceedings of the Sixth Applied Natural Language Processing Conference*, pages 224–231, 2000.

[10] E. Brill. Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part-of-Speech Tagging. *Computational Linguistics*, 21 (4):543–565, 1995.

[11] H. C. Carneiro, F. M. França, and P. M. Lima. Multilingual Part-of-speech Tagging with Weightless Neural Networks. *Neural Networks*, 66(C):11–21, 2015.

[12] G. Chrupala, G. Dinu, J. van Genabith, G. Chrupała, and J. van Genabith. Learning Morphology with Morfette. In *Proceedings of the 6th International Conference on Language Resources and Evaluation*, pages 2362–2367, 2008.

[13] M. Collins. Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1–8, 2002.

[14] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural Language Processing (Almost) from Scratch. *The Journal of Machine Learning Research*, 12:2493–2537, 2011.

[15] P. Compton and R. Jansen. A Philosophical Basis for Knowledge Acquisition. *Knowledge Aquisition*, 2(3): 241–257, 1990.

[16] A. Dalal, K. Nagaraj, U. Sawant, S. Shelke, and P. Bhattacharyya. Building Feature Rich POS Tagger for Morphologically Rich Languages: Experiences in Hindi. In *Proceedings of the 5th International Conference on Natural Language Processing*, 2007.

[17] P. Denis and B. Sagot. Coupling an Annotated Corpus and a Lexicon for State-of-the-art POS Tagging. *Language Resources and Evaluation*, 46:721–736, 2012.

[18] C. N. dos Santos, R. L. Milidiú, and R. P. Rentería. Portuguese Part-of-Speech Tagging Using Entropy Guided Transformation Learning. In *Proceedings of the 8th International Conference on the Computational Processing of Portuguese*, pages 143–152, 2008.

[19] R. Forsati and M. Shamsfard. Hybrid PoS-tagging: A Cooperation of Evolutionary and Statistical Approaches. *Applied Mathematical Modelling*, 38(13): 3193—3211, 2014.

[20] M. Fruzangohar, T. a. Kroeger, and D. L. Adelson. Improved Part-of-Speech Prediction in Suffix Analysis. *PloS one*, 8(10):e76042, 2013.

[21] C. Galves and P. Faria. Tycho Brahe Parsed Corpus of Historical Portuguese, 2010. URL http://www.tycho.iel.unicamp.br/$\sim$tycho.

[22] G. Georgiev, P. Nakov, P. Osenova, and K. Simov. Cross-lingual Adaptation as a Baseline : Adapting Maximum Entropy Models to Bulgarian. In *Proceedings of the Workshop Adaptation of Language Resources and Technology to New Domains 2009*, pages 35–38, 2009.

[23] G. Georgiev, V. Zhikov, P. Osenova, K. Simov, and P. Nakov. Feature-Rich Part-of-speech Tagging for Morphologically Complex Languages: Application to Bulgarian. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 492–502, 2012.

[24] 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*, pages 27–36, 2007.

[25] J. Giménez, L. Màrquez, and L. Marquez. SVMTool: A General POS Tagger Generator Based on Support Vector Machines. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, pages 43–46, 2004.

[26] T. Güngör. Part-of-Speech Tagging. In *Handbook of Natural Language Processing, second edition*, pages 205–235. Chapman & Hall/CRC, 2010.

[27] M. Hepple. Independence and Commitment: Assumptions for Rapid Training and Execution of Rule-based POS Taggers. In *Proceedings of 38th Annual Meeting of the Association for Computational Linguistics*, pages 277–278, 2000.

[28] T. Horsmann, N. Erbs, and T. Zesch. Fast or Accurate? – A Comparative Evaluation of PoS Tagging Models. In *Proceedings of the International Conference of the German Society for Computational Linguistics and Language Technology*, pages 22–30, 2015.

[29] N. Joshi, H. Darbari, and I. Mathur. HMM based POS tagger for Hindi. In *Proceedings of 2nd International Conference on Artificial Intelligence and Soft Computing*, pages 341–349, 2013.

[30] F. N. Kepler and M. Finger. Comparing Two Markov Methods for Part-of-speech Tagging of Portuguese. In *Proceedings of the 2nd International Joint Conference of IBERAMIA 2006 and SBIA 2006*, pages 482–491, 2006.

[31] Y.-B. Kim, B. Snyder, and R. Sarikaya. Part-of-speech Taggers for Low-resource Languages using CCA Features. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1292–1302, 2015.

[32] C. Kruengkrai, V. Sornlertlamvanich, and H. Isahara. A Conditional Random Field Framework for Thai Morphological Analysis. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation*, pages 2419–2424, 2006.

[33] M. Labeau, K. Löser, and A. Allauzen. Non-lexical Neural Architecture for Fine-grained POS Tagging. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 232–237, 2015.

[34] J. Lafferty, A. McCallum, and F. C. Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the*

18th International Conference on Machine Learning, pages 282–289, 2001.

[35] T. Lavergne, O. Cappé, and F. Yvon. Practical Very Large Scale CRFs. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 504–513, 2010.

[36] G. G. Lee, J.-H. Lee, and J. Cha. Syllable-pattern-based Unknown-morpheme Segmentation and Estimation for Hybrid Part-of-speech Tagging of Korean. *Computational Linguistics*, 28(1):53–70, 2002.

[37] Z. Li, J. Chao, M. Zhang, and W. Chen. Coupled Sequence Labeling on Heterogeneous Annotations: POS Tagging as a Case Study. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1783–1792, 2015.

[38] J. Ma, Y. Zhang, and J. Zhu. Tagging The Web: Building A Robust Web Tagger with Neural Network. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 144–154, 2014.

[39] Q. Ma, M. Murata, K. Uchimoto, and H. Isahara. Hybrid Neuro and Rule-Based Part of Speech Taggers. In *Proceedings of the 18th conference on Computational linguistics - Volume 1*, pages 509–515, 2000.

[40] M. P. Marcus, M. A. Marcinkiewicz, and B. Santorini. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993.

[41] M. Marimon, B. Fisas, N. Bel, M. Villegas, J. Vivaldi, S. Torner, M. Lorente, and S. Vázquez. The IULA Treebank. In *Proceedings of the eighth international conference on Language Resources and Evaluation*, pages 1920–1926, 2012.

[42] R. Moore. Fast High-Accuracy Part-of-Speech Tagging by Independent Classifiers. In *Proceedings the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1165–1176, 2014.

[43] T. Mueller, H. Schmid, and H. Schütze. Efficient Higher-Order CRFs for Morphological Tagging. In *Proceedings of the 2013 Conference on Empirical Methods on Natural Language Processing*, pages 322–332, 2013.

[44] T. Müller, R. Cotterell, A. Fraser, and H. Schütze. Joint Lemmatization and Morphological Tagging with Lemming. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2268–2274, 2015.

[45] M. Murata, Q. Ma, and H. Isahara. Comparison of Three Machine Learning Methods for Thai Part-of-Speech Tagging. *ACM Transactions on Asian Language Information Processing*, 1(2):145–158, 2002.

[46] G. Ngai and R. Florian. Transformation-Based Learning in the Fast Lane. In *Proceedings of the 2nd Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 1–8, 2001.

[47] D. Q. Nguyen, D. Q. Nguyen, S. B. Pham, and D. D. Pham. Ripple Down Rules for Part-of-Speech Tagging. In *Proceedings of the 12th International Conference on Intelligent Text Processing and Computational Linguistics - Volume Part I*, pages 190–201, 2011.

[48] D. Q. Nguyen, D. Q. Nguyen, D. D. Pham, and S. B. Pham. RDRPOSTagger: A Ripple Down Rules-based Part-Of-Speech Tagger. In *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 17–20, 2014.

[49] D. Q. Nguyen, D. Q. Nguyen, and S. B. Pham. Ripple Down Rules for Question Answering. *Semantic Web journal*, to appear, 2015. URL http://www.semantic-web-journal.net/.

[50] P. T. Nguyen, X. L. Vu, T. M. H. Nguyen, V. H. Nguyen, and H. P. Le. Building a Large Syntactically-Annotated Corpus of Vietnamese. In *Proceedings of the Third Linguistic Annotation Workshop*, pages 182–185, 2009.

[51] G. Noord, G. Bouma, F. Eynde, D. Kok, J. Linde, I. Schuurman, E. Sang, and V. Vandeghinste. Large Scale Syntactic Annotation of Written Dutch: Lassy. In *Essential Speech and Language Technology for Dutch*, Theory and Applications of Natural Language Processing, pages 147–164. 2013.

[52] N. Okazaki. CRFsuite: A Fast Implementation of Conditional Random Fields (CRFs), 2007. URL http://www.chokkan.org/software/crfsuite/.

[53] R. Östling. Stagger: an Open-Source Part of Speech Tagger for Swedish. *Northern European Journal of Language Technology*, 3:1–18, 2013.

[54] J. Pailai, R. Kongkachandra, T. Supnithi, and P. Boonkwan. A Comparative Study on Different Techniques for Thai Part-of-Speech Tagging. In *Proceedings of 10th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, pages 1–5, 2013.

[55] M. Palmer, R. Bhatt, B. Narasimhan, O. Rambow, D. M. Sharma, and F. Xia. Hindi Syntax: Annotating Dependency, Lexical Predicate-Argument Structure, and Phrase Structure. In *Proceedings of 7th International Conference on Natural Language Processing*, pages 261–268, 2009.

[56] A. Ratnaparkhi. A Maximum Entropy Model for Part-Of-Speech Tagging. In *Proceedings of the fourth Workshop on Very Large Corpora*, pages 133–142, 1996.

[57] D. Richards. Two Decades of Ripple Down Rules Research. *Knowledge Engineering Review*, 24(2):159–184, 2009.

[58] B. Sagot, L. Clément, E. V. d. L. Clergerie, and P. Boullier. The Lefff 2 Syntactic Lexicon for French: Architecture, Acquisition, Use. In *Proceedings of the 5th Language Resource and Evaluation Conference*, pages 1348–1351, 2006.

[59] C. D. Santos and B. Zadrozny. Learning Character-level Representations for Part-of-Speech Tagging. In *Proceedings of the 31st International Conference on Machine Learning*, pages 1818–1826, 2014.

[60] H. Schmid. Part-of-Speech Tagging with Neural Networks. In *Proceedings of the 15th International Conference on Computational Linguistics, Volume 1*, pages 172–176, 1994.

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

[62] H. Schmid and F. Laws. Estimation of Conditional Probabilities with Decision Trees and an Application to Fine-grained POS Tagging. In *Proceedings of 22nd International Conference on Computational Linguistics*, pages 777–784, 2008.

[63] T. Schnabel and H. Schütze. FLORS: Fast and Simple Domain Adaptation for Part-of-Speech Tagging. *Transactions of the Association for Computational Linguistics*, 2:15–26, 2014.

[64] I. Schroder. A Case Study in Part-of-Speech Tagging Using the ICOPOST Toolkit. Technical report, Department of Computer Science, University of Hamburg, 2002.

[65] D. Seddah, G. Chrupała, O. Cetinoglu, J. van Genabith, and M. Candito. Lemmatization and Lexicalized Statistical Parsing of Morphologically Rich Languages: the Case of French. In *Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 85–93, 2010.

[66] L. Shen, G. Satta, and A. Joshi. Guided Learning for Bidirectional Sequence Classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 760–767, 2007.

[67] K. Simov, P. Osenova, A. Simov, and M. Kouylekov. Design and Implementation of the Bulgarian HPSG-based Treebank. *Research on Language and Computation*, 2:495–522, 2004. URL http://www.bultreebank.org/btbmorf/.

[68] S. Singh, K. Gupta, M. Shrivastava, and P. Bhattacharyya. Morphological Richness Offsets Resource Demand- Experiences in Constructing a POS Tagger for Hindi. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 779–786, 2006.

[69] H.-J. Song, J.-W. Son, T.-G. Noh, S.-B. Park, and S.-J. Lee. A Cost Sensitive Part-of-Speech Tagging: Differentiating Serious Errors from Minor Errors. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1025–1034, 2012.

[70] V. Sornlertlamvanich, T. Charoenporn, and H. Isahara. ORCHID: Thai Part-Of-Speech Tagged Corpus, 1997. URL http://culturelab.in.th/files/orchid.html.

[71] D. j. Spoustová, J. Hajič, J. Raab, and M. Spousta. Semi-supervised Training for the Averaged Perceptron POS Tagger. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 763–771, 2009.

[72] SUC-3.0. The Stockholm—UmeåCorpus (SUC) 3.0, 2012. URL http://spraakbanken.gu.se/eng/resource/suc3.

[73] S. M. Thede and M. P. Harper. A Second-Order Hidden Markov Model for Part-of-Speech Tagging. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 175–182, 1999.

[74] K. Toutanova and C. D. Manning. Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-Speech Tagger. In *Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora*, pages 63–70, 2000.

[75] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer. Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics- Volume 1*, pages 173–180, 2003.

[76] O. T. Tran, C. A. Le, T. Q. Ha, and Q. H. Le. An Experimental Study on Vietnamese POS Tagging. In *Proceedings of 2009 International Conference on Asian Language Processing*, pages 23–27, 2009.

[77] Y. Tsuruoka and J. Tsujii. Bidirectional Inference with the Easiest-first Strategy for Tagging Sequence Data. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language*, pages 467–474, 2005.

[78] A. van den Bosch, B. Busser, S. Canisius, and W. Daelemans. An Efficient Memory-based Morphosyntactic Tagger and Parser for Dutch. In *Proceedings of the 17th Meeting of Computational Linguistics in the Netherlands*, pages 191–206, 2007.

[79] H. Xu and A. Hoffmann. RDRCE: Combining Machine Learning and Knowledge Acquisition. In *Proceedings of the 11th International Conference on Knowledge Management and Acquisition for Smart Systems and Services*, pages 165–179, 2010.

[80] X. Zheng, H. Chen, and T. Xu. Deep Learning for Chinese Word Segmentation and POS Tagging. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 647–657, 2013.