Disambiguating Main POS tags for Turkish

Razieh Ehsani
Istanbul Technical University
Faculty of Computer and Informatics
Computer Engineering
rehsani@itu.edu.tr

Muzaffer Ege Alper
National University of Singapore
Faculty of Science
Department of Statistics and Applied Probability
m.ege85@nus.edu.sg

Gülşen Eryiğit
Istanbul Technical University
Faculty of Computer and Informatics
Computer Engineering
gulsenc@itu.edu.tr

Eşref Adalı
Istanbul Technical University
Faculty of Computer and Informatics
Computer Engineering
adali@itu.edu.tr

[ ]

Abstract

This paper presents the results of main part-of-speech tagging of Turkish sentences using Conditional Random Fields (CRFs). Although CRFs are applied to many different languages for part-of-speech (POS) tagging, Turkish poses interesting challenges to be modeled with them. The challenges include issues related to the statistical model of the problem as well as issues related to computational complexity and scaling. In this paper, we propose a novel model for main-POS tagging in Turkish. Furthermore, we propose some approaches to reduce the computational complexity and allow better scaling characteristics or improve the performance without increased complexity. These approaches are discussed with respect to their advantages and disadvantages. We show that the best approach is competitive with the current state of the art in accuracy and also in training and test durations. The good results obtained imply a good first step towards full morphological disambiguation.

1 Introduction

The morphological disambiguation problem for morphologically rich languages differs significantly from the well known POS tagging problem. It is rather an automatic selection process from multiple legal analysis of a given word than the assignment of a POS tag from a predetermined tag set. The possible morphological analyses of a word (generally produced by a morphological analyzer) in such languages are very complex when compared to morphologically simple ones: They consist of the lemma, the main POS tags and the tags related to the inflectional and derivational affixes. The number of the set of possible morphological analyses may sometimes be infinite for some languages such as Turkish.

In this study, we focus on the determination of the main POS tags (which will be referred as "POS tagging" from now on) rather than the full disambiguation task. There are few methods for Turkish which directly tackle POS tagging problem. Instead many methods perform a full morphological disambiguation and the POS tags are obtained from the correct parses. In this work, we take a different approach and propose a model which directly tackles the POS tagging problem. While also being useful in its own right, this method is also a first step towards full morphological disambiguation through weighted opinion pooling approach [16].

To give a sense of the problem at hand and the general morphological disambiguation, we have measured the ambiguity corresponding to the POS tagging and Morphological Disambiguation problems. About 27% of the words in our corpus are ambiguous in terms of its POS tag and random guessing has an expected accuracy of 85%, on the other hand the ambiguity in terms of morphological disambiguation is about %50. The proposed approach in this paper improves the accuracy of POS tag to around 98.35%.

Our approach is based on the well known methodology of Conditional Random Fields, which is also applied to other languages with varying success. POS tagging problem was successfully tackled in languages with relatively simpler morphological properties (such as English) [16; 17; 6; 8]. On the other hand, other languages proved to be more problematic with lower tagging performance, [5; 15; 3] with accuracies ranging from %85 to %95. Smith et. al. [16] discusses the high computational burden of CRFs in both training and inference steps and argues that this is a major obstacle in its practical usage. In this work, we also discuss performance related issues and propose different approaches to lower the computational burden in inference step. The best approach among these approaches the state of the art [14] in performance, while being competitive in computational complexity. We also discuss the problem of feature selection in order to reduce training times and improve generalization capability. We employ the well known mRMR [13] method to this end. These efficiency improvements are important steps toward making CRFs more practical tools in NLP.

The paper is organized as follows, in Section 2 we discuss the properties of Turkish related to this paper. Next, a brief background on the statistical methodologies are given in Section 3. In Section 4, we introduce our approach to POS tagging together with a discussion of several methods to improve efficiency and performance of the basic method. Comparative results are given in the Experimental Results section and finally, we conclude in Section 6.

2 Turkish

Turkish is an agglunative language which has a complex morphological structure. This property of the Turkish language leads to vast amounts of different surface structures found in texts. In a corpus of ten million words, the number of distinct words exceeds four hundred thousand [10]. There are several suffixes, which may change the POS tags of the words from noun to verb or verb to adverb, etc. Thus, it is much harder to determine the final POS tag of a word using the root such as in English. Because of this, we can not resort to lexicons of words (roots) as in many studies on English. We must use the morphological analysis of the words to determine the tags. The context dependency of tags of words must also be taken into account.

There are several tags which determine respective properties of the associated words. These tags contain syntactic and semantic information and are called morphosyntactic or morphosemantic respectively. We use the same representation for the tags as [4]. Any words in Turkish can be represented by the chain of these tags. We call these chains of tags for words morphological analyses of these words.

Turkish morphological analysis considers 116 different tags. To better model these tags and circumvent the data sparseness problems, we have partitioned these into 9 disjoint groups, called slots. The slots are determined such that the semantic relation among the tags in a slot is maximum, while it is minimum for tags across slots. Also a word can not accept more than one tag from a single slot. Essentially transforming the problem into a multiple class classification problem. Such a construction of the problem, with this particular slot partitioning, is one of the contributions of the paper. The main properties of the words are expressed in the main POS category and the other slots serve to fill in the details such as plurality, tense, etc. In this paper, we are concerned with the correct disambiguation of the main POS tags, so we are interested in identifying the value of a single slot. However, the other slots serve as features in our models, which will be discussed in detail in later sections.

Many words in Turkish texts have more than one analysis. Sometimes the number of analyses reach 23. Because of the Turkish language derivative and inflective property, in theory, one word can use an infinite number of suffixes. Due to this, we are faced with immense vocabulary in Turkish. The large vocabulary size causes data sparseness problem. Some of these suffixes change the word meanings. In this case, these changes are expressed with inflectional groups (IGs) that are separated by ˆ*DB* sign, where ˆ*DB*'s mean derivation boundary (root+IG1+ ˆDB+IG2+ ˆDB+...+ ˆDB+IGn). One Turkish word can have many IGs in its analyzes. These IGs and the related tags can also be represented as tags. The standard morphological tags, also used in this work, are shown in Table 1. The example below shows the analyses for the word "alındı" produced by a Turkish two-level morphological analyzer [11].

1. al+Verb^DB+Verb+Pass+Pos+Past+A3sg (It was taken)
2. al+Adj^DB+Noun+Zero+A3sg+P2sg+Nom^DB+Verb+Zero+Past+A3sg (It was your red)
3. al+Adj^DB+Noun+Zero+A3sg+Pnon+Gen^DB+Verb+Zero+Past+A3sg (It was the one of the red)
4. alındı+Noun+A3sg+Pnon+Nom (receipt)
5. alın+Verb+Pos+Past+A3sg (resent)
6. alın+Noun+A3sg+Pnon+Nom^DB+Verb+Zero+Past+A3sg (It was the forhead)

Slot Groups	**Slot Values**
Main POS	Adj, Adv, Conj, Det, Dup, Interj, Noun, Num, Postp, Pron, Punc, Verb
Minor POS	Able, Acquire, ActOf, Adamantly, AfterDoingSo, Agt, Almost, As, AsIf, AsLongAs, Become, ByDoingSo, Card, Caus, DemonsP, Dim, Distrib, EverSince, FeelLike, FitFor, FutPart, Hastily, InBetween, Inf, Inf1, Inf2, Inf3, JustLike, Ly, Ness, NotState, Ord, Pass, PastPart, PCAbl, PCAcc, PCDat, PCGen, PCIns, PCNom, Percent, PersP, PresPart, Prop, Quant, QuesP, Range, Ratio,Real, Recip, ReflexP, Rel, Related, Repeat, Since, SinceDoingSo, Start, Stay, Time, When, While, With, Without, Zero
Person Agreements	A1pl, A1sg, A2pl, A2sg, A3pl, A3sg
Possessive Agreements	P1pl, P1sg, P2pl, P2sg, P3pl, P3sg, Pnon
Case Markers	Abl, Acc, Dat, Equ, Gen, Ins, Loc, Nom
Polarity	Neg, Pos
Tense/Mood	Aor, Desr, Fut, Imp, Neces, Opt, Pres, Prog1, Prog2, Cop, Cond, Past, Narr
Compund Tense	Comp_Cond, Comp_Narr, Comp_Past
Cop	Cop

Table 1: Morphological Tags

3 Background

In this section, we shall introduce the basic statistical mechanisms employed in this work. Discussion on details will be given in Section 4.

3.1 Conditional Random Fields

Simply put, CRF is a conditional distribution $p(\mathbf{y}|\mathbf{x})$ in the form of a Gibbs distribution and with an associated graphical structure encoding conditional independence assumptions. Because the model is conditional, dependencies among the input variables x are not explicitly represented, enabling the use of rich and global features of the input (neighboring words, capitalization…). CRFs are undirected graphical models used to calculate conditional probability of realizations of random variables on designated output nodes given the values assigned to other designed input nodes. In the special case, where the output nodes of the graphical model are linked by edges in a linear chain, CRFs make a first-order Markov independence assumption and thus can also be understood as a conditionally-trained finite state machine (FSM).

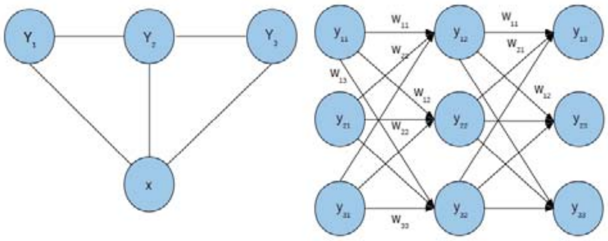


Figure 1: The equivalent expression of a linear chain CRF (on the left) as a FST (on the right)

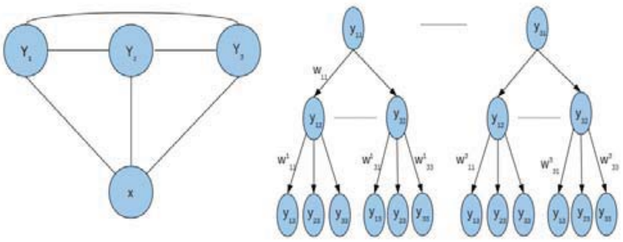


Figure 2: The equivalent expression of a 2nd order CRF (on the left) as a FST (on the right)

The distribution related to a given CRF is found using the normalized product of potential functions ($\Psi_C(\mathbf{y}_C)$) for each clique (C). The potential function itself can be, in principle, any non-negative function. Formally, the conditional probability $p(\mathbf{y}|\mathbf{x})$ can be expressed as

$$\begin{aligned} p(\mathbf{y}|\mathbf{x}) &= \tfrac{1}{Z(\mathbf{x})}\Pi_C \Psi_C(\mathbf{y}_C, \mathbf{x}) \\ &= \tfrac{1}{Z(\mathbf{x})} exp(-\textstyle\sum_C H_C(\mathbf{y}_C, \mathbf{x})) \end{aligned} \tag{1}$$

On the above equations, $H_C(\mathbf{y}_C, \mathbf{x}) = \log(\Psi_C(\mathbf{y}_C, \mathbf{x}))$. A CRF can also be seen as a weighted finite state transducer [16]. For example, in Figures 1 and 2, we can see the equivalent expression of a linear chain (1st order) CRF and 2nd order CRF as finite state transducers. These figures clearly show the parameter explosion when the order is increased. Higher number of parameters denies us the possibility of accurate parameter explosion in finite data. Indeed, using CRFs with order greater than one, deteriorates the model performance. On the other hand, a CRF of order 0 discards all neighbourhood information, effectively eliminating the advantages of sequential modeling. Unlike MEMM (see [7]), the transition weights in CRF are unnormalized, the weight of the whole path is normalized instead, which alleviates the label-bias problem.

The associated undirected graph of a CRF also indicates the conditional independence assumptions of the models. In undirected graphs, independence can be established simply by graph separation: if every path from a node in X to a node in Z goes through a node in Y , we conclude that $X \perp Z|Y$. In other words, X and Z are independent given Y. Properly modeling conditional independencies is essential in any statistical machine learning application, as having too many parameters will most often result in degraded performance.

3.2 Automatic Feature Selection

One method to improve the performance of a machine learning method is to select a subset of informative features [2]. The minimum Redundancy Maximum Relevance (mRMR [13]) method relies on the intuitive criteria for feature selection which states that the best feature set should give as much information regarding the class variable as possible while at the same time minimize inter-variable dependency as much as possible (avoiding redundancy). The two concepts, relevancy and redundancy, can be naturally expressed using information theoretic concept of mutual information. However, real data observed in various problems are usually too sparse to correctly estimate the joint probability distribution and consequently the full mutual information function. The solution proposed in [13], employs two different measures for redundancy (Red) and relevance (Rel):

$$Red = 1/|S|^2 \sum_{F_i, F_j \in S} MI(F_i, F_j) Rel = 1/|S| \sum_{F_i \in S} MI(F_i, R) \tag{2}$$

In the expressions above, S is the set of features of interest, $MI(.,.)$ is the mutual information function, R is the class variable and F_i is the random variable corresponding to the ith feature. Then the goal of mRMR is to select a feature set S that is as relevant ($\max(Rel)$) and as non redundant ($\min(Red)$) as possible. In the original work [13], two criteria to combine Rel and Red were proposed. In this work, the criterion of Mutual Information Difference ($MID = Rel - Red$) is used, because it is known to be more stable than the other proposed criterion ($MIQ = Rel/Red$) [1].

As a side note, we have also considered the "feature induction" in [8]. However, we have observed a significant drop in accuracy and therefore will not discuss this approach in this paper.

4 Proposed Framework

In the proposed method, POS tagging of a sentence is performed in a series of steps. In the most basic form we begin by computing the features related to the sentence, later the conditional probabilities of possible tag assignments are computed and the most probable tag sequence are selected.

The proposed method makes use of the mallet library [9] and the mRMR source code found in [12].

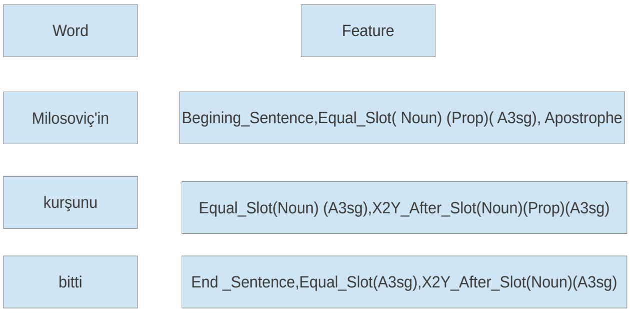
4.1 Features

In a linear chain conditional random field, there are two types of features, edge features and node features. Edge features are functions of labels of consecutive words ($f_k(y_i, y_{i+1})$) and node features are functions of words in the sentence ($f_k(y_i, \mathbf{x})$, where **x** denotes words of the sentence). The probability of a sequence is determined by the feature values as well as the associated model parameters. Thus, determining good feature functions that describe the important characteristics of the words is crucial for a successful model. We employ several morphological/syntactical properties as features.

In our model, the feature functions f_k are determined using several tests such as capitalization, end of sentence, etc. Results of these tests together constitute the features vector $F = f_1, f_2, ..., f_k$ for a word.

To illustrate the two kinds of features, let's consider one feature for node and edge type features used in our model. The *Color* feature is an example for a node feature, it is a function that returns one if the word is among a set of words describing colors and zero otherwise. The indicator function $\Phi(y_i = Adj, y_{i+1} = Noun)$, which returns one if the expression is true and zero otherwise, is an example of an edge feature.

The edge functions in our proposed method consist of all possible slot value pairs. The node functions are given in Table 2. The features "Color Set Feature", "Digit Set Feature", "Pronoun Set Feature", "Transition Set Feature" and "Non-Restrictive Set Feature" indicate whether the word is a member of corresponding sets of special words. These sets correspond to specific linguistic classes in Turkish language. The "Noun Adj Feature" indicates whether the word has suffixes that are generally used to change a noun to an adjective. "Capital Feature" indicates whether the word starts with a capital letter. "Before amount feature" and "Before Ques Morpheme Feature" indicate whether the word is followed by a special word/class of words. As their names imply, "Beginning Sentence Feature" and "End Sentence Feature" indicate whether the word is at the beginning or the end of the sentence. Finally, "Equal Slot", "X2Y Before" and "X2Y After" feature



Word	Feature
Milosoviç'in	Begining_Sentence,Equal_Slot(Noun) (Prop)(A3sg), Apostrophe
kurşunu	Equal_Slot(Noun) (A3sg),X2Y_After_Slot(Noun)(Prop)(A3sg)
bitti	End _Sentence,Equal_Slot(A3sg),X2Y_After_Slot(Noun)(A3sg)

Figure 3: A sample sentence and the corresponding features

templates generate features based on whether respectively the word itself, the word before or after it has a particular slot value which is unambiguously known, i.e. these values are the same for all possible analyses of the word. These classes of features contain 363 feature functions. However, in application, some of these features were discarded using mRMR as explained in Section 3.2. Figure 3 shows a sample sentence and the corresponding features. In this Figure, we observe that the first word ”Milosevic’in” gets the ”Beginning” feature. Since the morphological analyzer states that the fact that this word is ”A3sg”, ”Noun” and ”Prop” unambigously, i.e. these tags showup in all of the possible parses, we also have the ”Equal Slot” generated features of ”A3sg”, ”Noun” and ”Prop”. Finally, we see the feature ”X2Y Before A3sg” which means the word after this one is unambigously known to be ”A3sg”. We can confirm this by checking the next word ”kursunu” where we can see the feature ”A3sg” as expected. The features for the other words can be understood similarly.

Feature Templates	**Number of Corresponding Features**
Capital_Feature	1
End_Sentence_Feature	1
Begining_Sentence_Feature	1
Color_Set_Feature	1
Equal_Slot_Feature	116
Digit_Set_Feature	1
Before_Mi_Feature	1
Pronoun_Set_Feature	1
Transition_Set_Feature	1
Nonrestrictive_Set_Feature	1
Before_Amount_Feature	1
Noun_Adj_Feature	1
X2Y_Before_slot	116
X2Y_After_slot	116
After_Capital_Feature	1
Proper_Feature	1
PostP_Feature	1
Apostrophe_Feature	1
Total	363

Table 2: The features considered in this work

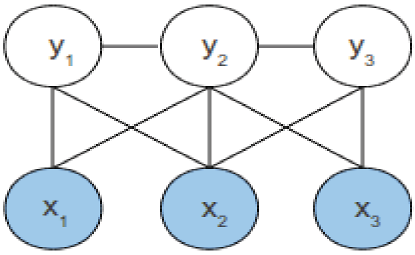


Figure 4: The graphical model of the proposed approach

4.2 Models

In this section, we explain our basic approach for POS tagging and introduce some slight variations which improve the efficiency and performance.

4.2.1 Basic Model

The CRF trained for POS tags are conditioned on the features of the sentence. However, during POS tagging, we also know a set of possible tags given by the morphological analyzer, which we call possible solution sequences ($\mathbf{S}_i$). Thus, we have a further conditioning.

$$p(\mathbf{S}_i|C) = \frac{p(\mathbf{S}_i|\mathcal{F}(C))}{\sum_j p(\mathbf{S}_j|\mathcal{F}(C))} \tag{3}$$

Where C is the sentence and $\mathcal{F}(C)$ is the corresponding feature representation of the sentence, given by the CRF. In other words, we do not assign the most probable tag sequence according to the conditional probability given by the CRF but select the most probable sequence ($\hat{\mathbf{t}}$) among possible sequences instead. This selection is performed by a constrained Viterbi approach, where the Viterbi is run on states that are deemed possible by the morphological analyser, instead of running Viterbi on the whole state space.

$$\hat{\mathbf{t}} = \arg\max_{S_i} p(\mathbf{S}_i|C) \tag{4}$$

The graphical model for the proposed method is shown in Figure 4.

Figure 5 shows a sample sentence and how our method chooses the POS tags. The top part of the figure shows the features for the respective words and the bottom part shows the possible POS tags as given by the analyzer. The values indicated above the arrows show transition weights. Note that in this example, any path from a tag of the initial word to a tag of the last word is a possible solution. In this figure, the weights of the transitions are the functions of the initial state, the final state and the features of the final word. The weight function is actually a factored expression, where $f(s_i, s_{i+1}, \mathcal{F}(w_{i+1})) = q(s_i, s_{i+1})q(s_{i+1}, \mathcal{F}(w_{i+1}))$, the first term corresponds to the edge features and the second term corresponds to node features.

4.2.2 Alternative Models

The basic approach of using CRF for POS tagging has an important disadvantage: high computational complexity. To remedy this issue, we propose these methods: dividing sentences into shorter sub-sentences and using marginal probabilities of tag assignments per word to eliminate the unlikely tags. In addition, we introduce a new approach to improve the performance of the basic method without significant overhead. In this section, we describe these methods and briefly comment on their performances. The quantitative results will be given in the Results Section.

Note that the complexity of the constrained Viterbi is $O(T \times |S|^2)$, where T is the length of the sequence and $|S|$ is the maximum number of possible states in any element of the sequence.

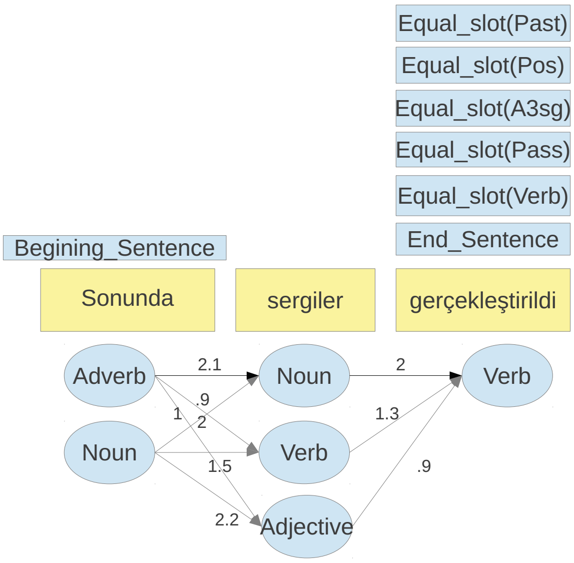


Figure 5: A sample sentence ("The exhibition has been finally realized.") with features and possible solutions. The tag chosen by our method is shown in bold arrows.

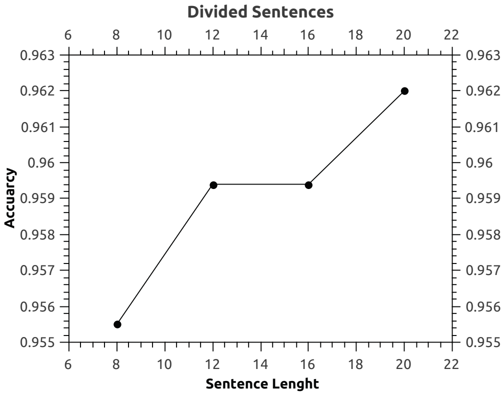


Figure 6: Accuracy vs. the length of the partial sentences

4.2.3 Model I: Splitting Sentences

This fast approximation method is conceptually the easiest one. The idea is to split a long sentence into multiple parts such that each part is shorter than a maximum length. Let's explain this method with an example sentence from our corpus. This sentence has 35389440 different possible morphological analysis sequences. The poor performance that would result from computing the probabilities of all of these possible solutions is obvious. Now suppose we divide the sentence into 4 parts of lengths 9,9,9,7. The corresponding number of possible solutions are 384, 960, 30 and 32 which sum up to 1406. The huge savings in the number of solutions to consider is apparent. However, despite these good reductions in the number of possible solutions to consider, this method results in the worst accuracy among the alternatives. This is due to the fact that splitting sentences this way enforces an independence assumption on the splitted sub-sentences, which reduces the performance especially in words that are closer to the cut-off boundaries. The Figure 6 shows the tradeoff between the performance and the length of the partial sentences.

Using this approach, the complexity of disambiguating a sentence is reduced to $O(T' \times |S|^2)$, where T'

is the maximum length of the sub-sentences, so the reduction is linear.

4.2.4 Model II: Trim Unlikely Tags

Notice that the compexity of the constrained Viterbi is linear on the length but quadratic on the maximum number of states for any element of the sequence. This observation becomes even more important when we note that the number of possible analysis of a word can reach up to 23 in our corpus and possibly more in general texts. Thus a reduction on the number of possible tag assigments of a word can have significant effects. Out of the many possible sequences for the sentence mentioned in Section 4.2.3, many include highly unlikely values for some words. The approach discussed in this section exploits this pattern by trimming out the highly unlikely tags for words but still allowing multiple possible POS tags. In our implementation, we select the words for which the number of possible tag assignments is greater than 6. For such words, we remove the least likely tag assignments using marginal probabilities until either this number is 6 or the number of eliminated tags is 5. We use such an upper limit in order not to remove too many such tags in order not to degrade accuracy. The additional complexity of this approach is obviously linear on the length of the sequence and the trimmed sequence can be disambiguated by constrained Viterbi in $O(T \times 6) = O(T)$. We can see that there can be huge savings in long sentences with compex morphological properties. The conservative approach outlined here means the accuracy is not effected at all, as shown in the next section.

4.2.5 Model III: Model Complexity of the Solutions

An interesting observation of morphological properties of words in Turkish is that the correct POS tags of the words tend to be the less morphologically complex ones. In other words, simpler interpretations of words tend to be used more often than the more complex ones of the same word. One way to operationalise this observation is to take the Bayesian stance and model a prior. However, correctly assigning numerical values for our prior knowledge is difficult and we take the other position, where the nature of this relation is learned from the data itself. In Turkish, the morphological complexity of a word can be modeled by the number of IGs of it. Thus we model this number with a 0-order CRF, since we do not expect the neighbouring IG counts to effect each other. This CRF is combined with the original one by multiplying the probabilities, i.e. we assume the number of IGs and the POS tags to be independent, which is reasonable. Since we use a 0-order CRF, the compexity of inference is only $O(T \times |S|)$. However, we do note increased performance as can be seen in the next section.

5 Experimental Results

In this section, we first show the effect of feature selection on the performance. We then show the performance of the proposed method on a common dataset and compare it with the method of [14], which is considered as the state of art. The results are obtained using default parameters of the mallet library. The Java source codes used in the experiments will be made available online.

5.1 POS tagging Results

The results for the proposed method, together with the results from [14] (Perceptron) are given in Table 3. We use the same training data (1 million words) that is used in these studies. The training data is a semi-automatically tagged data set which consists some erroneous analyses. In this study, we strived to correct as many errors as possible and trained our methods as well as the previous methods on this dataset. We have also accounted to the difference in tags employed in Hasim Sak's method and ours so we kept two separate training files, each having the same corrections but slightly different tags, so that Hasim Sak's method does not suffer from the changes in some of the tag names. Our test data (a manually disambiguated data consisting nearly 1K words) is again from [18]. Note that this set also contains errenous analyses, which we had to correct. All the results are reported using this corrected dataset, which will be made available to

Method	test set
Perc [14]	98.60
Basic Model	98.35
Model I	96.2
Model II	98.35
Model III	98.48
Model II + Model III	98.48

Table 3: Pos Tagging Performances

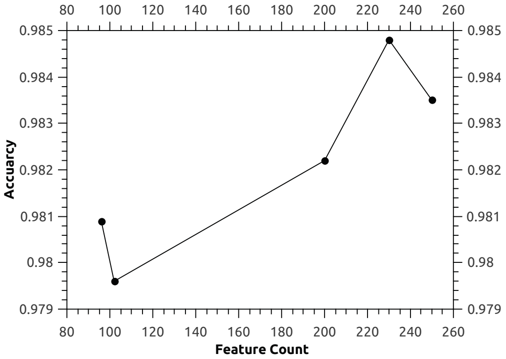


Figure 7: Accuracy vs. number of features selected by mRMR

researchers. These corrections are the reason why our results are slightly different than the ones reported in [14] The results are reported in Table 3.

The results in Table 3 exclude the punctuations in computing the accuracy. The results indicate the competitiveness of our approach. It is important to recognize that the POS tagging in Perceptron [14] method is performed by selecting the appropriate tags after a full morphological disambiguation. On the contrary, our method directly assigns a POS tag sequence to the sentence. The output of our method need not be a single assignment, instead we can output different "belief levels" for different tag assignments. If these POS tags are to be used in another procedure as an intermediate step, this will also be an advantage. Finally, the method in [14] contains a lot more number of features than our proposed approach, since our approach is flexible in the selection features, it can be extended using additional features from the Perceptron method.

5.2 Automatic Feature Selection Results

Feature selection is an important step in many machine learning tasks. The effect of feature selection is two-folds, the reduction of features may actually increase classification performance, since accidental correlations in the training data can mislead the classifier and generalization capability of classifiers is expected to be better for lower model complexity. Another effect is the improvement in training and classification efficiency, since inference in the model with a fewer number of features will be faster. For these reasons, we have dismissed the features that are not selected in the top 230 by mRMR.

Figure 7 shows the accuracy vs. the number of features. We can see that reducing the features below 230 degrades the performance significantly. Even though a significant increase in performance is not observed for the particular validation set, the reduction in features is still relevant to reduce computational complexity in test and training.

6 Conclusions

In this paper, we proposed a method using Conditional Random Fields to solve the problem of POS tagging in Turkish. We have shown that using several features derived from morphological and syntactic properties of words and feature selection, we were able to achieve a performance competitive to the state of art. Furthermore, the probabilistic nature of our method makes it possible for it to be utilized as an intermediate step in another NLP task, such that the belief distribution can be used as a whole instead of a single estimate. Note that our proposed method can also be employed to other languages, perhaps with the addition of language dependent features.

Another major contribution of this work is the discussion on several approaches to improve efficiency of POS tagging using CRFs. We believe this work constitutes a major step towards making CRF a more practical tool in NLP.

As part of our future work, we plan to investigate the addition of other features to improve the performance of the proposed method. One possibility is to incorporate features based on lemma. Eventually, we plan to combine several CRF models to solve the full disambiguation task, which poses several interesting challenges.

References

[1] Gokhan Gulgezen, Zehra Cataltepe, and Lei Yu. Stable and Accurate Feature Selection. In Wray Buntine, Marko Grobelnik, Dunja Mladenić, and John Shawe-Taylor, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 5781, chapter 47, pages 455–468. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[2] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182, March 2003.

[3] Nizar Habash and Owen Rambow. Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 573–580, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

[4] Dilek Hakkani-Tür, Kemal Oflazer, and Gökhan Tür. Statistical morphological disambiguation for agglutinative languages. *Computers and the Humanities*, 36(4):381–410, 2002.

[5] T. Kudo, K. Yamamoto, and Y. Matsumoto. Applying conditional random fields to japanese morphological analysis. In *Proc. of EMNLP*, volume 2004, 2004.

[6] J. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. 2001.

[7] Andrew McCallum, Dayne Freitag, and Fernando C. N. Pereira. Maximum entropy markov models for information extraction and segmentation. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, pages 591–598, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[8] Andrew McCallum and Wei Li. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 188–191, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[9] Andrew Kachites McCallum. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu, 2002.

[10] Erhan Mengusoglu and Olivier Deroo. Turkish lvcsr: Database preparation and language modeling for an agglutinative language. In *in ICASSPâ2001, Student Forum, Salt-Lake City*, 2001.

[11] K. Oflazer. Two-level description of turkish morphology. *Literary and Linguistic Computing*, 9(2):137–148, April 1995.

[12] Hanchuan Peng. mrmr (minimum redundancy maximum relevance feature selection), 2012.

[13] Hanchuan Peng, Fuhui Long, and C. Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(8):1226 –1238, aug. 2005.

[14] Hasim Sak, Tunga Gungor, and Murat Saraclar. Morphological disambiguation of turkish text with perceptron algorithm. In *Proceedings of the 8th International Conference on Computational Linguistics and Intelligent Text Processing*, CICLing '07, pages 107–118, Berlin, Heidelberg, 2007. Springer-Verlag.

[15] D. Shacham and S. Wintner. Morphological disambiguation of hebrew: A case study in classifier combination. In *Proceedings of EMNLP-CoNLL*, volume 7, pages 439–447, 2007.

[16] Noah A. Smith, David A. Smith, and Roy W. Tromble. Context-based morphological disambiguation with random fields. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 475–482, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

[17] C. Sutton and A. McCallum. An introduction to conditional random fields. *Arxiv preprint arXiv:1011.4088*, 2010.

[18] Deniz Yuret and Ferhan Töre. Learning morphological disambiguation rules for turkish. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, pages 328–334, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.