Learning Morphological Disambiguation Rules for Turkish

Deniz Yuret
Dept. of Computer Engineering
Koç University
İstanbul, Turkey
dyuret@ku.edu.tr

Ferhan Türe
Dept. of Computer Engineering
Koç University
İstanbul, Turkey
fture@ku.edu.tr

Abstract

In this paper, we present a rule based model for morphological disambiguation of Turkish. The rules are generated by a novel decision list learning algorithm using supervised training. Morphological ambiguity (e.g. lives = live+s or life+s) is a challenging problem for agglutinative languages like Turkish where close to half of the words in running text are morphologically ambiguous. Furthermore, it is possible for a word to take an unlimited number of suffixes, therefore the number of possible morphological tags is unlimited. We attempted to cope with these problems by training a separate model for each of the 126 morphological features recognized by the morphological analyzer. The resulting decision lists independently vote on each of the potential parses of a word and the final parse is selected based on our confidence on these votes. The accuracy of our model (96%) is slightly above the best previously reported results which use statistical models. For comparison, when we train a single decision list on full tags instead of using separate models on each feature we get 91% accuracy.

1 Introduction

Morphological disambiguation is the task of selecting the correct morphological parse for a given word in a given context. The possible parses of a word are generated by a morphological analyzer. In Turkish, close to half the words in running text are morphologically ambiguous. Below is a typical word "masalı" with three possible parses.

masal+Noun+A3sg+Pnon+Acc (= *the story*)
masal+Noun+A3sg+P3sg+Nom (= *his story*)
masa+Noun+A3sg+Pnon+Nomˆ DB+Adj+With (= *with tables*)

Table 1: Three parses of the word "masalı"

The first two parses start with the same root, masal (= *story, fable*), but the interpretation of the following +ı suffix is the Accusative marker in one case, and third person possessive agreement in the other. The third parse starts with a different root, masa (= *table*) followed by a derivational suffix +lı (= *with*) which turns the noun into an adjective. The symbol ˆDB represents a derivational boundary and splits the parse into chunks called inflectional groups (IGs).[1]

We will use the term *feature* to refer to individual morphological features like +Acc and +With; the term *IG* to refer to groups of features split by derivational boundaries (ˆDB), and the term *tag* to refer to the sequence of IGs following the root.

Morphological disambiguation is a useful first step for higher level analysis of any language but it is especially critical for agglutinative languages like Turkish, Czech, Hungarian, and Finnish. These languages have a relatively free constituent order, and

[1] See (Oflazer et al., 1999) for a detailed description of the morphological features used in this paper.

Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL, pages 328–334,
New York, June 2006.

syntactic relations are partly determined by morphological features. Many applications including syntactic parsing, word sense disambiguation, text to speech synthesis and spelling correction depend on accurate analyses of words.

An important qualitative difference between part of speech tagging in English and morphological disambiguation in an agglutinative language like Turkish is the number of possible tags that can be assigned to a word. Typical English tag sets include less than a hundred tag types representing syntactic and morphological information. The number of potential morphological tags in Turkish is theoretically unlimited. We have observed more than ten thousand tag types in our training corpus of a million words. The high number of possible tags poses a data sparseness challenge for the typical machine learning approach, somewhat akin to what we observe in word sense disambiguation.

One way out of this dilemma could be to ignore the detailed morphological structure of the word and focus on determining only the major and minor parts of speech. However (Oflazer et al., 1999) observes that the modifier words in Turkish can have dependencies to any one of the inflectional groups of a derived word. For example, in "mavi masalı oda" (= *the room with a blue table*) the adjective "mavi" (= *blue*) modifies the noun root "masa" (= *table*) even though the final part of speech of "masalı" is an adjective. Therefore, the final part of speech and inflection of a word do not carry sufficient information for the identification of the syntactic dependencies it is involved in. One needs the full morphological analysis.

Our approach to the data sparseness problem is to consider each morphological feature separately. Even though the number of potential tags is unlimited, the number of morphological features is small: The Turkish morphological analyzer we use (Oflazer, 1994) produces tags that consist of 126 unique features. For each unique feature f, we take the subset of the training data in which one of the parses for each instance contain f. We then split this subset into positive and negative examples depending on whether the correct parse contains the feature f. These examples are used to learn rules using the Greedy Prepend Algorithm (GPA), a novel decision list learner.

To predict the tag of an unknown word, first the morphological analyzer is used to generate all its possible parses. The decision lists are then used to predict the presence or absence of each of the features contained in the candidate parses. The results are probabilistically combined taking into account the accuracy of each decision list to select the best parse. The resulting tagging accuracy is 96% on a hand tagged test set.

A more direct approach would be to train a single decision list using the full tags as the target classification. Given a word in context, such a decision list assigns a complete morphological tag instead of predicting individual morphological features. As such, it does not need the output of a morphological analyzer and should be considered a tagger rather than a disambiguator. For comparison, such a decision list was built, and its accuracy was determined to be 91% on the same test set.

The main reason we chose to work with decision lists and the GPA algorithm is their robustness to irrelevant or redundant features. The input to the decision lists include the suffixes of all possible lengths and character type information within a five word window. Each instance ends up with 40 attributes on average which are highly redundant and mostly irrelevant. GPA is able to sort out the relevant features automatically and build a fairly accurate model. Our experiments with Naive Bayes resulted in a significantly worse performance. Typical statistical approaches include the tags of the previous words as inputs in the model. GPA was able to deliver good performance without using the previous tags as inputs, because it was able to extract equivalent information implicit in the surface attributes. Finally, unlike most statistical approaches, the resulting models of GPA are human readable and open to interpretation as Section 3.1 illustrates.

The next section will review related work. Section 3 introduces decision lists and the GPA training algorithm. Section 4 presents the experiments and the results.

2 Related Work

There is a large body of work on morphological disambiguation and part of speech tagging using a variety of rule-based and statistical approaches. In the

rule-based approach a large number of hand crafted rules are used to select the correct morphological parse or POS tag of a given word in a given context (Karlsson et al., 1995; Oflazer and Tür, 1997). In the statistical approach a hand tagged corpus is used to train a probabilistic model which is then used to select the best tags in unseen text (Church, 1988; Hakkani-Tür et al., 2002). Examples of statistical and machine learning approaches that have been used for tagging include transformation based learning (Brill, 1995), memory based learning (Daelemans et al., 1996), and maximum entropy models (Ratnaparkhi, 1996). It is also possible to train statistical models using unlabeled data with the expectation maximization algorithm (Cutting et al., 1992). Van Halteren (1999) gives a comprehensive overview of syntactic word-class tagging.

Previous work on morphological disambiguation of inflectional or agglutinative languages include unsupervised learning for of Hebrew (Levinger et al., 1995), maximum entropy modeling for Czech (Hajič and Hladká, 1998), combination of statistical and rule-based disambiguation methods for Basque (Ezeiza et al., 1998), transformation based tagging for Hungarian (Megyesi, 1999).

Early work on Turkish used a constraint-based approach with hand crafted rules (Oflazer and Kuruöz, 1994). A purely statistical morphological disambiguation model was recently introduced (Hakkani-Tür et al., 2002). To counter the data sparseness problem the morphological parses are split across their derivational boundaries and certain independence assumptions are made in the prediction of each inflectional group.

A combination of three ideas makes our approach unique in the field: (1) the use of decision lists and a novel learning algorithm that combine the statistical and rule based techniques, (2) the treatment of each individual feature separately to address the data sparseness problem, and (3) the lack of dependence on previous tags and relying on surface attributes alone.

3 Decision Lists

We introduce a new method for morphological disambiguation based on decision lists. A decision list is an ordered list of rules where each rule consists of a pattern and a classification (Rivest, 1987). In our application the pattern specifies the surface attributes of the words surrounding the target such as suffixes and character types (e.g. upper vs. lower case, use of punctuation, digits). The classification indicates the presence or absence of a morphological feature for the center word.

3.1 A Sample Decision List

We will explain the rules and their patterns using the sample decision list in Table 2 trained to identify the feature `+Det` (determiner).

Rule	Class	Pattern
1	1	W=˜çok R1=+DA
2	1	L1=˜pek
3	0	W=+AzI
4	0	W=˜çok
5	1	–

Table 2: A five rule decision list for `+Det`

The value in the class column is 1 if word W should have a `+Det` feature and 0 otherwise. The pattern column describes the required attributes of the words surrounding the target word for the rule to match. The last (default) rule has no pattern, matches every instance, and assigns them `+Det`. This default rule captures the behavior of the majority of the training instances which had `+Det` in their correct parse. Rule 4 indicates a common exception: the frequently used word "çok" (meaning very) should not be assigned `+Det` by default: "çok" can be also used as an adjective, an adverb, or a postposition. Rule 1 introduces an exception to rule 4: if the right neighbor R1 ends with the suffix +DA (the locative suffix) then "çok" should receive `+Det`. The meanings of various symbols in the patterns are described below.

When the decision list is applied to a window of words, the rules are tried in the order from the most specific (rule 1) to the most general (rule 5). The first rule that matches is used to predict the classification of the center word. The last rule acts as a catch-all; if none of the other rules have matched, this rule assigns the instance a default classification. For example, the five rule decision list given above classifies the middle word in "pek çok alanda" (matches rule

W	target word	A	[ae]
L1, L2	left neighbors	I	[ıiuü]
R1, R2	right neighbors	D	[dt]
==	exact match	B	[bp]
=~	case insensitive match	C	[cç]
=+	is a suffix of	K	[kgğ]

Table 3: Symbols used in the rule patterns. Capital letters on the right represent character groups useful in identifying phonetic variations of certain suffixes, e.g. the locative suffix +DA can surface as +de, +da, +te, or +ta depending on the root word ending.

1) and "pek çok insan" (matches rule 2) as `+Det`, but "insan çok daha" (matches rule 4) as not `+Det`.

One way to interpret a decision list is as a sequence of *if-then-else* constructs familiar from programming languages. Another way is to see the last rule as the default classification, the previous rule as specifying a set of exceptions to the default, the rule before that as specifying exceptions to these exceptions and so on.

3.2 The Greedy Prepend Algorithm (GPA)

To learn a decision list from a given set of training examples the general approach is to start with a default rule or an empty decision list and keep adding the best rule to cover the unclassified or misclassified examples. The new rules can be added to the end of the list (Clark and Niblett, 1989), the front of the list (Webb and Brkic, 1993), or other positions (Newlands and Webb, 2004). Other design decisions include the criteria used to select the "best rule" and how to search for it.

The Greedy Prepend Algorithm (GPA) is a variant of the PREPEND algorithm (Webb and Brkic, 1993). It starts with a default rule that matches all instances and classifies them using the most common class in the training data. Then it keeps prepending the rule with the maximum *gain* to the front of the growing decision list until no further improvement can be made. The algorithm can be described as follows:

GPA($data$)

```
1  dlist ← NIL
2  default-class ← MOST-COMMON-CLASS(data)
3  rule ← [if TRUE then default-class]
4  while GAIN(rule, dlist, data) > 0
5      do dlist ← prepend(rule, dlist)
6         rule ← MAX-GAIN-RULE(dlist, data)
7  return dlist
```

The gain of a candidate rule in GPA is defined as the increase in the number of correctly classified instances in the training set as a result of prepending the rule to the existing decision list. This is in contrast with the original PREPEND algorithm which uses the less direct Laplace preference function (Webb and Brkic, 1993; Clark and Boswell, 1991).

To find the next rule with the maximum gain, GPA uses a heuristic search algorithm. Candidate rules are generated by adding a single new attribute to the pattern of each rule already in the decision list. The candidate with the maximum gain is prepended to the decision list and the process is repeated until no more positive gain rules can be found. Note that if the best possible rule has more than one extra attribute compared to the existing rules in the decision list, a suboptimal rule will be selected. The original PREPEND uses an admissible search algorithm, OPUS, which is guaranteed to find the best possible candidate (Webb, 1995), but we found OPUS to be too slow to be practical for a problem of this scale.

We picked GPA for the morphological disambiguation problem because we find it to be fast and fairly robust to the existence of irrelevant or redundant attributes. The average training instance has 40 attributes describing the suffixes of all possible lengths and character type information in a five word window. Most of this information is redundant or irrelevant to the problem at hand. The number of distinct attributes is on the order of the number of distinct word-forms in the training set. Nevertheless GPA is able to process a million training instances for each of the 126 unique morphological features and produce a model with state of the art accuracy in about two hours on a regular desktop PC.[2]

[2]Pentium 4 CPU 2.40GHz

4 Experiments and Results

In this section we present the details of the data, the training and testing procedures, the surface attributes used, and the accuracy results.

4.1 Training Data

documents	2383
sentences	50673
tokens	948404
parses	1.76 per token
IGs	1.33 per parse
features	3.29 per IG
unique tokens	111467
unique tags	11084
unique IGs	2440
unique features	126
ambiguous tokens	399223 (42.1%)

Table 4: Statistics for the training data

Our training data consists of about 1 million words of semi-automatically disambiguated Turkish news text. For each one of the 126 unique morphological features, we used the subset of the training data in which instances have the given feature in at least one of their generated parses. We then split this subset into positive and negative examples depending on whether the correct parse contains the given feature. A decision list specific to that feature is created using GPA based on these examples.

Some relevant statistics for the training data are given in Table 4.

4.2 Input Attributes

Once the training data is selected for a particular morphological feature, each instance is represented by surface attributes of five words centered around the target word. We have tried larger window sizes but no significant improvement was observed. The attributes computed for each word in the window consist of the following:

1. The exact word string (e.g. W==Ali'nin)
2. The lowercase version (e.g. W=˜ali'nin) Note: all digits are replaced by 0's at this stage.
3. All suffixes of the lowercase version (e.g. W=+n, W=+In, W=+nIn, W=+'nIn, etc.) Note: certain characters are replaced with capital letters representing character groups mentioned in Table 3. These groups help the algorithm recognize different forms of a suffix created by the phonetic rules of Turkish: for example the locative suffix +DA can surface as +de, +da, +te, or +ta depending on the ending of the root word.
4. Attributes indicating the types of characters at various positions of the word (e.g. Ali'nin would be described with W=UPPER-FIRST, W=LOWER-MID, W=APOS-MID, W=LOWER-LAST)

Each training instance is represented by 40 attributes on average. The GPA procedure is responsible for picking the attributes that are relevant to the decision. No dictionary information is required or used, therefore the models are fairly robust to unknown words. One potentially useful source of attributes is the tags assigned to previous words which we plan to experiment with in future work.

4.3 The Decision Lists

At the conclusion of the training, 126 decision lists are produced of the form given in Table 2. The number of rules in each decision list range from 1 to 6145. The longer decision lists are typically for part of speech features, e.g. distinguishing nouns from adjectives, and contain rules specific to lexical items. The average number of rules is 266. To get an estimate on the accuracy of each decision list, we split the one million word data into training, validation, and test portions using the ratio 4:1:1. The training set accuracy of the decision lists is consistently above 98%. The test set accuracies of the 126 decision lists range from 80% to 100% with the average at 95%. Table 5 gives the six worst features with test set accuracy below 89%; these are the most difficult to disambiguate.

4.4 Correct Tag Selection

To evaluate the candidate tags, we need to combine the results of the decision lists. We assume that the presence or absence of each feature is an independent event with a probability determined by the test set accuracy of the corresponding decision list. For example, if the `+P3pl` decision list predicts YES, we assume that the `+P3pl` feature is present with

87.89%	+Acquire	To acquire (noun)
86.18%	+PCIns	Postposition subcat.
85.11%	+Fut	Future tense
84.08%	+P3pl	3. plural possessive
80.79%	+Neces	Must
79.81%	+Become	To become (noun)

Table 5: The six features with the worst test set accuracy.

probability 0.8408 (See Table 5). If the +Fut decision list predicts NO, we assume the +Fut feature is present with probability $1 - 0.8511 = 0.1489$. To avoid zero probabilities we cap the test set accuracies at 99%.

Each candidate tag indicates the presence of certain features and the absence of others. The probability of the tag being correct under our independence assumption is the product of the probabilities for the presence and absence of each of the 126 features as determined by our decision lists. For efficiency, one can neglect the features that are absent from all the candidate tags because their contribution will not effect the comparison.

4.5 Results

The final evaluation of the model was performed on a test data set of 958 instances. The possible parses for each instance were generated by the morphological analyzer and the correct one was picked manually. 40% of the instances were ambiguous, which on the average had 3.9 parses. The disambiguation accuracy of our model was 95.82%. The 95% confidence interval for the accuracy is [0.9457, 0.9708].

An analysis of the mistakes in the test data show that at least some of them are due to incorrect tags in our training data. The training data was semi-automatically generated and thus contained some errors. Based on hand evaluation of the differences between the training data tags and the GPA generated tags, we estimate the accuracy of the training data to be below 95%. We ran two further experiments to see if we could improve on the initial results.

In our first experiment we used our original model to re-tag the training data. The re-tagged training data was used to construct a new model. The resulting accuracy on the test set increased to 96.03%, not a statistically significant improvement.

In our second experiment we used only unambiguous instances for training. Decision list training requires negative examples, so we selected random unambiguous instances for positive and negative examples for each feature. The accuracy of the resulting model on the test set was 82.57%. The problem with selecting unambiguous instances is that certain common disambiguation decisions are never represented during training. More careful selection of negative examples and a sophisticated bootstrapping mechanism may still make this approach workable.

Finally, we decided to see if our decision lists could be used for tagging rather than disambiguation, i.e. given a word in a context decide on the full tag *without the help of a morphological analyzer*. Even though the number of possible tags is unlimited, the most frequent 1000 tags cover about 99% of the instances. A single decision list trained with the full tags was able to achieve 91.23% accuracy using 10000 rules. This is a promising result and will be explored further in future work.

5 Contributions

We have presented an automated approach to learn morphological disambiguation rules for Turkish using a novel decision list induction algorithm, GPA. The only input to the rules are the surface attributes of a five word window. The approach can be generalized to other agglutinative languages which share the common challenge of a large number of potential tags. Our approach for resolving the data sparseness problem caused by the large number of tags is to generate a separate model for each morphological feature. The predictions for individual features are probabilistically combined based on the accuracy of each model to select the best tag. We were able to achieve an accuracy around 96% using this approach.

Acknowledgments

We would like to thank Kemal Oflazer of Sabancı University for providing us with the Turkish morphological analyzer, training and testing data for disambiguation, and valuable feedback.

References

Brill, E. (1995). Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics*, 21(4):543–565.

Church, K. W. (1988). A stochastic parts program and noun phrase parser for unrestricted text. In *Proceedings of the Second Conference on Applied Natural Language Processing*, pages 136–143.

Clark, P. and Boswell, R. (1991). Rule induction with CN2: Some recent improvements. In Kodratoff, Y., editor, *Machine Learning – Proceedings of the Fifth European Conference (EWSL-91)*, pages 151–163, Berlin. Springer-Verlag.

Clark, P. and Niblett, T. (1989). The CN2 induction algorithm. *Machine Learning*, 3:261–283.

Cutting, D., Kupiec, J., Pedersen, J., and Sibun, P. (1992). A practical part-of-speech tagger. In *Proceedings of the 3rd Conference on Applied Language Processing*, pages 133–140.

Daelemans, W. et al. (1996). MBT: A memory-based part of speech tagger-generator. In Ejerhead, E. and Dagan, I., editors, *Proceedings of the Fourth Workshop on Very Large Corpora*, pages 14–27.

Ezeiza, N. et al. (1998). Combining stochastic and rule-based methods for disambiguation in agglutinative languages. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics (COLING/ACL98)*, pages 379–384.

Hajič, J. and Hladká, B. (1998). Tagging inflective languages: Prediction of morphological categories for a rich, structured tagset. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics (COLING/ACL98)*, pages 483–490, Montreal, Canada.

Hakkani-Tür, D. Z., Oflazer, K., and Tür, G. (2002). Statistical morphological disambiguation for agglutinative languages. *Computers and the Humanities*, 36:381–410.

Karlsson, F., Voutialinen, A., Heikkilä, J., and Anttila, A. (1995). *Constraint Grammar - A Language Independent System for Parsing Unrestricted Text*. Mouton de Gruyter.

Levinger, M., Ornan, U., and Itai, A. (1995). Learning morpho-lexical probabilities from an untagged corpus with an application to hebrew. *Computational Linguistics*, 21(3):383–404.

Megyesi, B. (1999). Improving brill's pos tagger for an agglutinative language. In Pascale, F. and Joe, Z., editors, *Proceedings of the Joing SIGDAT Conference on Empirical Methods in Natural Language and Very Large Corpora*, pages 275–284, College Park, Maryland, USA.

Newlands, D. and Webb, G. I. (2004). Alternative strategies for decision list construction. In *Proceedings of the Fourth Data Mining Conference (DM IV 03)*, pages 265–273.

Oflazer, K. (1994). Two-level description of turkish morphology. *Literary and Linguistic Computing*, 9(2):137–148.

Oflazer, K., Hakkani-Tür, D. Z., and Tür, G. (1999). Design for a turkish treebank. In *Proceedings of the Workshop on Linguistically Interpreted Corpora, EACL 99*, Bergen, Norway.

Oflazer, K. and Kuruöz, İ. (1994). Tagging and morphological disambiguation of turkish text. In *Proceedings of the 4th Applied Natural Language Processing Conference*, pages 144–149. ACL.

Oflazer, K. and Tür, G. (1997). Morphological disambiguation by voting constraints. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics (ACL97, EACL97)*, Madrid, Spain.

Ratnaparkhi, A. (1996). A maximum entropy model for part-of-speech tagging. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Rivest, R. L. (1987). Learning decision lists. *Machine Learning*, 2:229–246.

van Halteren, H., editor (1999). *Syntactic Wordclass Tagging*. Text, Speech and Language Technology. Kluwer Academic Publishers.

Webb, G. I. (1995). Opus: An efficient admissible algorithm for unordered search. *JAIR*, 3:431–465.

Webb, G. I. and Brkic, N. (1993). Learning decision lists by prepending inferred rules. In *Proceedings of the AI 93 Workshop on Machine Learning and Hybrid Systems*, pages 6–10, Melbourne.