TRmorph: A morphological analyzer for Turkish

Çağrı Çöltekin

Draft: July 29, 2013

A word of warning: This document describes the new/development version of TRmorph. As such, there may be some mismatches between what is documented here and how the analyzer behaves. This version is a complete overwrite of the previous version reported in Çöltekin 2010. If you are using the older version (you shouldn't), this document is probably useless for you.

1 Introduction

TRmorph is an open-source[1] finite-state morphological analyzer for Turkish. The official web site of the analyzer is `http://www.let.rug.nl/coltekin/trmorph`. Since it's release and introduction in Çöltekin 2010, it has been used by many researchers and NLP practitioners. This document gives a brief description of how to use the tools that comes with this package, as well as some implementation details that may be helpful for people customizing this open-source tool for their own needs.

Earlier versions of trmorph was implemented using SFST (Schmid 2005), current version is implemented more wide-spread finite state description languages *lexc* and *xfst* from Xerox (Beesley and Karttunen 2003), using Foma (Hulden 2009) as the main development tool.

The lexc/xfst implementation of TRmorph should compile with any lexc/xfst compiler without much additional effort. The only foma-specific notation used in the morphology description is about handling duplication, which can also be handled with twolc rules, or compile-replace (Beesley and Karttunen 2003).[2]

[1] Current version of TRmorph is licensed under GNU Lesser General Public License. See README file in the TRmorph distribution for more information.

[2] Compilation with HFST tools (Lindén et al. 2009) without modification is not a problem since HFST uses foma as back end for parsing xfst files.

2 How to use it

2.1 Compilation from the source

To compile TRmorph from sources, besides a lexc/xfst compiler such as foma, you will need a C preprocessor, GNU make on a UNIX-like system for compiling TRmorph.

If you meet all requirements to build analyzer/generator FST, you should type `make` in the main TRmorph distribution to get the compiled automaton file `trmorph.fst`.

Trmorph also comes with a set of other finite-state tools that can be useful in various tasks. Currently, tolls for the following is distributed together with TRmorph.

- stemming/lemmatization
- morphological segmentation
- hyphenation
- guessing unknown words

To compile these tools, you should type specify the FST you want to build as an argument to `make`, e.g., `make stemmer` will build an binary automaton called `stem.fst`. These additional tools are described in Section 5.

2.2 Customizing TRmorph

TRmorph is an open source utility. As a result, you are free to modify the source according to your needs. Source code includes some useful comments on what/how/where things are done. Furthermore, TRmorph can be customized for some common choices during the compilation. These options are typically related to more relaxed analysis. For example whether to allow non-capitalized proper names, or analyze (and generate) text written in all capitals, or set the decimal and thousand separator in numbers. These options are set in file `options.h`. The file contains necessary documentation, and new options are being added based on user requests.

2.3 Trying it out

Assuming you build the analyzer `trmorph.fst` using foma, you can simply start foma and use xfst commands implemented in foma to analyze and generate the words. Here is an example session:

```
$ foma
foma[0]: regex @"trmorph.fst";
2.1 MB. 62236 states, 135237 arcs, Cyclic.
foma[1]: up okudum
oku<V><past><1s>
foma[1]: down oku<V><past><1s>
okudum
```

The first line is from shell starting foma, the command to foma on the second line reads file `trmorph.fst`. The fourth line asks for analysis of the verb *okudum* 'I read-PAST' and the fifth line is the output of the analysis. The sixth line asks for the generation of the analysis string obtained, and not surprisingly the result is the same.

Note that some of the output is suppressed for readability, and this is one of the rare cases where the analysis is unambiguous. Turkish morphological analysis is an ambiguous process, and TRmorph does not try to avoid it during the analysis.[3]

Once you are convinced that the output may be useful for your purposes, you will probably want to use it analyzing large amount of text. For batch analysis tasks foma's `flookup` utility may fit better.

To use the analyzer with HFST, you need to compile the source automaton with HFST tools.

[3] For most purposes, the output of the morphological analyzer has little use without disambiguation. There are quite a few disambiguators reported in the literature, but, as yet, there are no disambiguators that work with TRmorph output.

3 The tagset

Since the first SFST version, the tagset of TRmorph evolved considerably, mostly as response to user requests. The description of the morphology mostly follows Göksel and Kerslake 2005. However, there are frequent divergences. This section describes the tags used in TRmorph. The aim of this section is to help users understand the output of the system. Occasional discussion of the morphological process is also included, but this section documents neither the morphology of the language nor the way it is implemented in TRmorph. The index at the end of the document also allows easy access to the points where a particular tag is defined or mentioned in this document.

A clarification of the notation for the surface forms is in order before starting the documentation of the tagset and related suffixes. Suffixes in Turkish typically contain under-specified vowels and consonants that are resolved according to morphophonological rules, like vowel harmony. These vowels and consonants are indicated with capital letters listed below.

A is realized as either 'a' or 'e'.

I is realized as either 'ı', 'i', 'u' or 'ü'.

D is realized as either 'd' or 't'.

P is realized as either 'p' or 'b'.

K is realized as either 'k', 'ğ' or 'y'.

C is realized as either 'c' or 'ç'.

A letter in parentheses indicate a buffer consonant or vowel, that may be dropped in certain contexts.

3.1 Part-of-speech tags

All words at least one POS tag. A word may get multiple POS tags, and even the same tag repetitively. This tracks the changes to POS during suffixation. This process follows the idea of *inflectional groups* (IGs, Oflazer,

Hakkani-Tür, et al. 1999; Oflazer, Say, et al. 2003), that are typically used in Turkish NLP literature. The idea is that a word in Turkish may start with a particular part of speech due to lexical properties of the root, may get some inflectional suffixes that modify this POS, then may get a derivational suffix which changes its POS, and may again take additional inflections. This process may continue theoretically indefinitely. Logically, the POS tag of the final word is the last POS tag in the sequence. However, the other intermediate forms are also important, and, at least in typical dependency parsing, can participate in dependency relations with other words by themselves. For example the analysis of the word *evdekilerinki*[4] in example 1.

(1) ev⟨N⟩⟨loc⟩⟨ki⟩⟨Adj⟩⟨0⟩⟨N⟩⟨pl⟩⟨gen⟩⟨ki⟩⟨Adj⟩

The example analysis in (1) includes the following IGs

1. The initial noun 'ev' with locative maker.
2. Addition of the suffix -ki makes an adjective.
3. The adjective becomes a (pro)nominal with a zero derivation, which takes plural suffix and genitive marker.
4. Yet another -ki is suffixed, and the word becomes an adjective again.[5]

TRmorph, by default, does not mark IG boundaries explicitly. However, one can easily trace the IG changes following the POS tags. All POS tag names start with a capital letter, while other tags always start with a lowercase letter or number.

All part-of speech tags used in TRmorph are listed in Table 1.

Most POS tags are self explanatory, and does not require much explanation. The following part of speech tags are somewhat unusual and deserves some explanation.

⟨Exist⟩ is used for two words *var* 'existent/present' and *yok* 'non-existent/absent', where the latter is marked as ⟨neg⟩. These words behave mostly like nouns in their predicate function (with zero copula), but marking them simply as nouns does blur their function.

[4]Rough translation: 'the *ones* that belong to the *ones* in the house', as in 'the *books* that belong to the *people* in the house'.

[5]More likely reading of this example includes another zero derivation causing final POS to be again noun.

Table 1: The list of part of speech tags in TRmorph along with the corresponding tags in some of the output produced by in Oflazer's (1994) morphological analyzer.

Tag	Description	Oflazer's tag
⟨Alpha⟩	Symbols of alphabet	?
⟨Adj⟩	Adjective	+Adj
⟨Adv⟩	Adverb	+Adverb
⟨Cnj⟩	Conjunction	+Conjunction
⟨Det⟩	Determiner	+Det
⟨Exist⟩	*var* and *yok*	+Adj
⟨Ij⟩	Interjection	+Interj
⟨N⟩	Noun	+Noun
⟨Not⟩	*değil*	+Verb
⟨Num⟩	Number	+Num
⟨Onom⟩	Onomatopoeia	?
⟨Postp⟩	Postposition	+Postp
⟨Prn⟩	Pronoun	+Pron
⟨Punc⟩	Punctuation	+Punc
⟨Q⟩	Question particle	+Ques, also +QuesP
⟨V⟩	Verb	+Verb

⟨Not⟩ is used for *değil* 'not' only. Like *var* and *yok*, *değil* also behaves like nominal predicates. But again, marking it as noun or verb hides the fact that it has a special function.

⟨Q⟩ is used for the question particle -mI. The question particle is written separately from the predicate it modifies. However, the preferred analysis of question particle in TRmorph is together with the predicate. This ensures that it follows a correct form of the predicate and vowel harmony is applied correctly. However, since we do not assume that the input is tokenized with this assumption, this form make sure that the input is analyzed.

This tag is related to the lowercase ⟨q⟩ tag, which is used if input is tokenized such that the predicate and the question particle form a single token. The question words corresponding to wh-words in English are marked with the corresponding POS tag together with the tag ⟨qst⟩.

3.2 Subcategorization of lexemes

Besides the major major POS tags or word classes discussed above, we often we assign subcategories to a lexeme. Typically the subcategorization is applied to a root form in the lexicon, but some morphemes and POS tags after a derivation may also receive a subcategory. Subcategories defined here are features of a morpheme that do not have a surface realization. Representing these using a different notation allows to make this distinction, and a surface–analysis mapping that is (almost) one-to-one. However, these tags can be viewed as any other feature of a word (or IG), and the notation can be replaced with more uniform notation easily.

The subcategories generally mark semantic differences, but they may also result in morphosyntactic differences. Lexical subcategorization in TRmorph output is marked using the syntax ⟨`Cat:subcat`$_1$`:subcat`$_2$`:…`⟩, where 'Cat' is a major category and 'subcat$_1$', 'subcat$_2$' and so on are sub categories. A typical example of a subcategory is the *proper* nouns, which are tagged as ⟨`N:prop`⟩.

The following lists subcategories in used in TRmorph for all POS tags that may receive a subcategory.

Nouns besides the tag ⟨`N:prop`⟩ marking proper names, abbreviated nouns are marked with the tag ⟨`N:abbr`⟩. If the lexeme is an abbreviated proper name, then the tag would be ⟨`N:prop:abbr`⟩.

Conjunctions are subcategorized as *coordinating*, *adverbial* or *subordinating* conjunctions, marked using tags ⟨`Cnj:coo`⟩, ⟨`Cnj:adv`⟩, ⟨`Cnj:sub`⟩ respectively.

The last one of these categories, ⟨`Cnj:sub`⟩, include only a limited set of conjunctions which come first in a subordinate clause. These words currently are *ki*, *eğer* and *şayet*, all borrowings from Persian. The other subordinating particles/words occur at the end of the subordinate clause, and they are marked as postpositions (⟨`Postp`⟩) described below. Furthermore, most of the subordination is in Turkish is done through suffixation which is described in Section 3.14.

Pronouns Pronouns are further categorized as *personal*, *demonstrative* and *locative* pronouns, marked using ⟨`Prn:pers`⟩, ⟨`Prn:dem`⟩, ⟨`Prn:locp`⟩ respectively. Furthermore, the pronouns that question that form questions, like *kim* 'who', and *ne* 'what', are marked as ⟨`Prn:qst`⟩. Like in conjunctions both subcategory markers can be present. For example *kim* 'who' would be marked as ⟨`Prn:pers:qst`⟩.

Besides the above subcategories, pronouns get number and person agreement markers. These markers can be useful in subject-predicate agreement as well as some other constructions (such as genitive-possessive construction involving pronouns). However, the agreement in Turkish is far from being trivial (see Göksel and Kerslake 2005, pp.116–122). The markers ⟨`Prn:pers:1s`⟩, ⟨`Prn:pers:2s`⟩, ⟨`Prn:pers:3s`⟩, ⟨`Prn:pers:1p`⟩, ⟨`Prn:pers:2p`⟩ and ⟨`Prn:pers:3p`⟩ are tags used for the person and number the pronominal will agree. The reflexive pronoun *kendi(si)* is marked as ⟨`Prn:pers:refl`⟩

Subcategorization of pronouns, particularly as personal pronouns, are sometimes not a clear decision. Subcategories of pronouns are left unspecified even though they are primarily used as personal pronouns, and some pronoun marked as personal pronouns may refer to entities other than people.

Determiners are marked based on their definiteness. *Demonstrative* determiners are marked ⟨`Det:def`⟩ and indefinite determiners are marked ⟨`Det:indef`⟩. The question words that take place of determiners *ne kadar* 'how much' and *hangi* 'which' are tagged with ⟨`Det:qst`⟩.

Further subcategorization of determiners (for example quantifiers) can be implemented in the future.

Postpositions are always subcategorized in two dimensions. First subcategory is the category of the resulting postpositional phrase, either an *adjectival* or *adverbial* phrase, marked as ⟨`Postp:adj`⟩ and ⟨`Postp:adv`⟩ respectively.

Postpositions choose the noun phrase complements. Besides the category of the resulting phrase, postpositions also include a tag specifying the requirement for the complement noun phrase. The complement requirement tag is formed a concise description of

the requirement followed by capital letter 'C'. The postpositions that require the complement to be in *ablative*, *dative* and *instrumental* cases are marked ⟨Postp:ablC⟩, ⟨Postp:datC⟩, and ⟨Postp:insC⟩ respectively. Postpositions that require non-case marked complement receives ⟨Postp:nomC⟩. The postpositions that require the noun phrase to be suffixes with either -Il or -sIz are marked with ⟨Postp:liC⟩.[6] Finally, postpositions that require a number as their complement are marked with ⟨Postp:numC⟩

Numbers are tagged as ⟨Num:ara⟩ for Arabic numerals, and ⟨Num:rom⟩ for Roman numerals. Numbers that are spelled out are not marked with a subcategory marker (but still marked as ⟨Num⟩). Besides numbers the question word *kaç* 'how many' is also tagged as a number with a sub tag specifying that it is a question word, resulting in ⟨Num:qst⟩.

Verbs are currently not subcategorized in TRmorph.

Subcategorizing verbs *transitive* and *intransitive*, or marking all types (cases) of noun phrase complements a verb can take is planned and some early steps are underway as of this writing (July 2013).

Adverbs are not currently subcategorized, except a few adverbial question words for which the tag ⟨Adv:qst⟩ is used.

Exist The tag ⟨Exist⟩ exists only for two words *var* 'existent/present' and *yok* 'non-existent/absent'. Since *yok* is the negative of *var*, it is tagged as negative: ⟨Exist:neg⟩.

Some verbs, nouns, adjectives, adverbs and conjunctions are formed from more than one written words. Some of these are local, like the adverb *apar topar* 'hurriedly', but some may be split like the conjunction *ya ya evdedir ya iş yerinde* 's/he is either at home or the office'. Furthermore, some of individual 'words' in such constructions cannot be used by themselves, like *topar* above. If the non-split multi-word expressions are input to the analyzer together, they are analyzes like other words of the same class. However, if they are input word-by-word a sub tag ⟨:partial⟩ is added to the main POS tag. For example *apar* and *topar* is be tagged as ⟨Adv:partial⟩ and *ya* is tagged as ⟨Cnj:partial⟩ (more precisely ⟨Cnj:coo:partial⟩). Similarly, currently the tags ⟨N:partial⟩, ⟨Adj:partial⟩ and ⟨V:partial⟩ are used for nouns, adjectives and verbs respectively.

[6] Like -(y)lA that we mark as ⟨ins⟩, these suffixes are typically considered derivational suffixes, however their use resemble case markers.

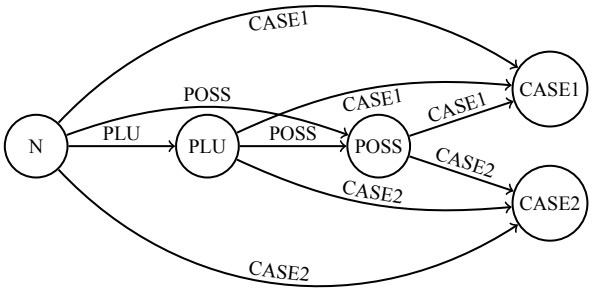

Figure 1: Automata depicting noun inflections. The edge CASE1 represents the locative and ablative suffixes, CASE2 represents all other case-like suffixes. The reason for for differentiation is due to the fact that the state CASE1 can be followed by -ki.

3.3 Nominal morphology and noun inflections

Nouns, pronouns, adjectives and adverbs in Turkish are considered nominals. Most adverbs and adjectives can function as nouns (or pronouns). For example *mavi* 'blue' may mean 'the blue one', or *hızlı* 'fast' may mean 'the fast one'. In TRmorph this is handled by allowing any adjective or adverb to become an noun with a *zero derivation*. A zero derivation is always marked with tag ⟨0⟩ and followed by the new POS tag, in this case ⟨N⟩.

Nouns can receive plural suffix, one of the possessive suffixes and one of the case suffixes. All of these inflections are optional. However, when they co-occur, they have to occur in this order. Full list of noun inflections are presented in Table 2.

If there is a plural marker analysis symbol after the ⟨N⟩ will include a ⟨pl⟩. TRmorph does not mark for singular. If a noun is not marked for plural, it is assumed to be singular.

Table 2: Noun inflections.

Function		surface	tag
	Plural	-lAr	⟨pl⟩
Possessive	First person singular	-(I)m	⟨p1s⟩
	Second person singular	-(I)n	⟨p2s⟩
	Third person singular	-(s)I	⟨p3s⟩
	First person plural	-(I)mIz	⟨p1p⟩
	Second person plural	-(I)nIz	⟨p2p⟩
	Third person plural	-lArI	⟨p3p⟩
Case	Accusative	-(y)I	⟨acc⟩
	Dative	-(y)A	⟨dat⟩
	Ablative	-DAn	⟨abl⟩
	Locative	-DA	⟨loc⟩
	Genitive	-(n)In	⟨gen⟩
	Instrumental/commutative	-(y)lA	⟨ins⟩

Possessive markers follow either nominal stem, or the plural marker. Besides marking for possession, these suffixes derive pronouns when attached to adjectivals(e.g., determiners or adjectives).

The first five cases in Table 2 are commonly recognized cases in Turkish. The *instrumental/commutative* marker also behaves like case suffixes. There are two more suffixes -lI and -sIz (resembling to abessive case in its function), that can occupy the same slot, which are marked with tags ⟨li⟩ and ⟨siz⟩ respectively.

The -(s)I suffix, listed as ⟨p3s⟩ in Table 2, is highly ambiguous. One of its many functions that can be confused with the possessive suffix is forming noun compounds. TRmorph marks this usage of the suffix -(s)I as ⟨ncomp⟩. In this use, this suffix always causes ambiguities. Besides the fact that a noun suffixed with -(s)I can be either marked for possession or head of a noun compound, since one of the two -(s)I suffixes following each other is deleted from the surface, it can be both (a noun compound marked for possession).

The case (or case-like) suffixes change the role of the noun (or the noun phrase headed by the noun) in the sentence. For example a locative marked noun phrase may function as an adverb (*saat dokuz*da *görüşürüz*) or an adjective (*yedi yaşında çocuk*). However, following the common practice in the literature we do not attempt to mark possible POS changes after case-like markers.

3.4 The suffix -ki

The suffix -ki, tagged as ⟨ki⟩, attaches to a locative or ablative marked nouns. The resulting word functions as an adjective or pronoun. In both cases, TRmorph marks the transition to adjective. For example, *evdeki* is analyzed as 'ev⟨N⟩⟨loc⟩⟨ki⟩⟨Adj⟩'. Since all adjectives are allowed to become a noun through a zero derivation, the pronoun reading is intended to be represented by this change. For example, the intended analysis for *evdeki kitap* 'the book in the house' is 'ev⟨N⟩⟨loc⟩⟨ki⟩⟨Adj⟩', while analysis for *evdeki uyuyor* 'the one/person in the house is sleeping' suffixes '⟨0⟩⟨N⟩' to this analysis.

The (pro)noun formed by -ki can further be suffixed with other nominal suffixes. Although the number of iterations using multiple -ki suffixes are limited in practice, there is no principle limit, causing in-principle unbounded length for a word in Turkish.

3.5 Tags related to nominal predicates

Any nominal in Turkish may become a predicate by attaching one of the copular suffixes -(y)DI, -(y)mIş, -(y)sA or -(y). These suffixes correspond to *past*, *evidential*, *conditional*, and *present* predicates involving the copula 'be'. The copular markers has to follow one of the verbal person agreement markers. For example *öğrenciy*di*k* 'We were students'), *öğrenciy*miş*ler* 'They were (evidentially) students'), *öğrenciy*se*n* 'If you are/were a student', *öğrenciyim* 'I'm a student'). Since the third person singular agreement suffix has no surface form and the buffer -(y)- does not surface in this case, any nominal without additional copular or person suffixes serve as a nominal predicate with present copula and third person singular agreement. Additionally, since a predicate with third person singular agreement also agrees with a third person plural agreement, we additionally mark such a noun as having present copula and third person plural agreement (for example, *babam öğretmen, annem ve ablam doktor* 'my father is a teacher, my mother and older sister are doctors').

TRmorph handles this process by allowing any noun to first became a verb by a zero derivation, and then mark-

ing them verb with the appropriate copula and the person agreement marker. The tags for copula are ⟨cpl:pres⟩, ⟨cpl:past⟩, ⟨cpl:evid⟩ and ⟨cpl:cond⟩ for , *present*, *past*, *evidential* and *conditional* copula respectively. Last three tags are also possible after a verb with a tense/aspect/modality suffix, and will be discussed further below. Example analyses (without root) for the examples above would be as follows:

öğrenciydik	⟨N⟩⟨0⟩⟨V⟩⟨cpl:past⟩⟨1p⟩
öğrenciymişler	⟨N⟩⟨0⟩⟨V⟩⟨cpl:evid⟩⟨3p⟩
öğrenciysen	⟨N⟩⟨0⟩⟨V⟩⟨cpl:cond⟩⟨2s⟩
öğrenciyim	⟨N⟩⟨0⟩⟨V⟩⟨cpl:pres⟩⟨1s⟩
öğretmen	⟨N⟩⟨0⟩⟨V⟩⟨cpl:pres⟩⟨3s⟩
doktor	⟨N⟩⟨0⟩⟨V⟩⟨cpl:pres⟩⟨3p⟩

Besides copular suffixes, the suffix -(y)ken (making adverbials from verbs, discussed below) may occupy the same slot as the copular suffixes.

The nominal predicate with a copula and person agreement may be followed by marker Göksel and Kerslake 2005 call 'generalizing modality marker', the suffix -DIr. It is particularly common with ⟨3s⟩ case. The tag for this marker in TRmorph is ⟨dir⟩.

3.6 Number inflections

The suffix -(ş)Ar, tagged ⟨dist⟩, attached to numbers form *distributive* numerals. Besides the numbers (written as numerals or spelled out), question word *kaç* 'how many' may also get this suffix, and tagged with ⟨dist⟩.

The ordinal numerals are formed using the suffix –(I)ncI, and tagged as ⟨ord⟩.

Percent sign before numerals are treated like a prefix, and tagged as ⟨perc⟩.

3.7 Verbal voice suffixes

Turkish verbs can be marked as *reflexive*, *reciprocal*, *causative* and *passive* voice. The tags used for these functions are ⟨rfl⟩, ⟨rcp⟩, ⟨caus⟩ and ⟨pass⟩, respectively. The first two are rather unproductive while causative and passive forms are productive. Furthermore, causative suffix can be used repetitively.[7] With some verbs, double causative suffix has the same semantics as single suffix. TRmorph does not treat these cases separately. If surface string has double causative suffixes, the analysis will include two ⟨caus⟩ tags, regardless of its semantics.

Despite the fact that most grammar books list voice suffixes under inflectional morphology, TRmorph treats them as derivations, i.e., a ⟨V⟩ tag follows the voice related tags.

[7] Again, although this is limited in practice there is no principled limit on how many causative suffixes one can string on after another.

Table 3: Suffixes that make compound verbs.

Suffix	Tag	Expresses
-(y)AbIl	⟨abil⟩	ability
-(y)Iver	⟨iver⟩	immediacy
-(y)Agel	⟨agel⟩	habitual/long term
-(y)Adur	⟨adur⟩	repetition/continuity
-(y)Ayaz	⟨ayaz⟩	almost
-(y)Akal	⟨akal⟩	stop/freeze in action
-(y)Agör	⟨agor⟩	somewhat like ⟨iver⟩

3.8 Compound verbs

A verbal stem (possibly including voice suffixes) may be followed by a set of suffixes which are forms of other verbs. These suffixes are listed in Table 3.

The first three suffixes in this Table 3 are relatively productive, the others are rare or their use are lexicalized. Although rare, more than one these suffixes may attach to the same stem.

The form of ⟨abil⟩ in a negative verb is -(y)A, and unlike the rest of the suffixes listed in Table 3 it follows the negative marker.

Currently, these suffixes are not marked as derivations (causing an IG change).

3.9 Negative marker

Negation of a verbal predicate is indicated with the suffix -mA, and marked simply as ⟨neg⟩. For nominal predicates do not get this suffix, instead the particle *değil* is used.

3.10 Tense/aspect/modality markers

The verbal suffixes and tags described so far forms a non-finite verb. A verb with a set of suffixes described above

Table 4: Tense/aspect/modality markers. The usage of suffix -(y)A to express conditional aspect is informal, and rather restrictive. Aorist suffix is highly irregular. The choice of -Ar and -Ir depends on the stem. The -z form occurs only after negative marker, and it is not realized on the surface if it precedes first person agreement suffixes.

Tag	Suffix	Description
⟨evid⟩	-mIş	evidential past (perfective)
⟨fut⟩	-(y)AcAk	future
⟨obl⟩	-mAlI	obligative
⟨impf⟩	-mAktA	imperfective
⟨cont⟩	-(I)yor	imperfective
⟨past⟩	-DI	past (perfective)
⟨cond⟩	-sA,-(y)A,-	conditional
⟨opt⟩	-(y)A	optative
⟨imp⟩	-	imperative
⟨aor⟩	-Ar,-Ir,-z	aorist

Table 5: Verbal person agreement markers. The first character of the person agreement marker indicates the person, and second one indicates number (*s*ingular or *p*lural). The suffixes listed in the column column marked 'TAM1' follow the TAM markers ⟨evid⟩,⟨fut⟩,⟨obl⟩,⟨impf⟩ and ⟨cont⟩ as well as the evidential copula ⟨cpl:evid⟩ and nominal predicates. The same set of suffixes also follow positive verbs with ⟨aor⟩ without a negative marker. The suffixes on column marked 'TAM2' are used after ⟨past⟩ and ⟨cond⟩ as well as the corresponding copular markers ⟨cpl:past⟩ and ⟨cpl:cond⟩.

Tag	TAM1	TAM2	optative	imperative
⟨1s⟩	-(y)Im	-m	-(y)Im	*
⟨2s⟩	-sIn	-n	-sIn	-
⟨3s⟩	-	-	-	-sIn
⟨1p⟩	-(y)Iz	-K	-lIm	*
⟨2p⟩	-sInIz	-nIz	-sInIz	-(y)In,-(y)InIz
⟨3p⟩	-lAr	-lAr	-lAr	-sInlAr,-

either becomes a finite word by taking one of the tense, aspect and modality (TAM) markers or can be subject to subordination and becomes nominalized.

The list of TAM suffixes, the corresponding tags and brief descriptions are given in Table 4.

3.11 Person and number agreement

After TAM markers a finite verb requires one of the person and number agreement markers. The surface form of the person agreement markers change depending on the suffixes they follow. Table 5 lists the person agreement markers and their surface form according the TAM of the verb they attach to.

3.12 Copular markers and -DIr

The copular suffixes discussed in Section 3.5 can also be attached to a tensed verb, typically forming complex tenses. These suffixes are -(y)DI, -(y)mIş and -(y)sA, tagged as ⟨cpl:past⟩, ⟨cpl:evid⟩ and ⟨cpl:cond⟩, respectively.

The conditional copula -(y)sA can co-occur with other copular markers. When there is a copular suffix, person agreement suffixes normally attach after the first copula. However the third person plural suffix may be after the TAM marker or second copular suffix as well.

Similar to the nominal predicates with a copula, copular suffixes may follow the 'generalizing modality marker' -DIr tagged as ⟨dir⟩.

3.13 Question particle

Question particle -mI, tagged as ⟨Q⟩, is normally written separately. However, it has an intimate relationship between the verb or the nominal predicate it attaches to. First, it typically is attached to a tensed verb without a person agreement. In which case, the person agreement and the suffixes that may follow are attached to the question particle. Second, it follows the vowel harmony rules, and the underspecified vowel on -mI is realized based on the last vowel of the verb. As a result the question particle can only be analyzed (and generated) precisely together with the word it is attached to.

If tokenized together with the predicate, TRmorph will swallow the space in between the predicate and the -mI and analyze it altogether. In this case the tag ⟨q⟩ is used.

Furthermore, it is a common spelling mistakes to write the question particle together with the related word. TRmorph can instructed to to accept this common mistake during compile time, in which case the tag will again be ⟨q⟩.

3.14 Subordination

A set of suffixes attached to an 'untensed' verb, a verb without any TAM markers, results in the phrase headed by the verb to become a subordinate clause. TRmorph follows the description in Göksel and Kerslake 2005, and makes the distinction between three different forms of subordination. First, a set of suffixes produce *verbal noun*s from a non-finite verb. The resulting words function as nouns, and with some limitation they can receive all nominal inflections. The second group form *participles*, which form relative clauses. Participles can also take any of the nominal inflections with few restrictions. The last group, *converbs*, form adverbials and they are much more restricted in terms of the morphemes attached to them. The suffixes that for any of these nominal forms overlap significantly.

TRmorph uses the tag structure ⟨type:subtype⟩ for marking subordinating suffixes. The first part, type, is one of vn, part and cv for verbal nouns, participles and converbs, respectively. The second, subtype, part indicate a further distinction of the function of the suffix, a relevant linguistic abbreviation, but sometimes a version of the surface form of the suffix. The tags used for all three types of subordinating suffixes are listed in Table 6.

Since verbal nouns participles and converbs derive nominal, adjectival and adverbial phrases, respectively, POS tags, ⟨N⟩, ⟨Adj⟩ and ⟨Adv⟩, follow these tags.

Some of the suffixes have multiple functions and may derive more than one type of subordinated clauses. Furthermore TRmorph will have some spurious ambiguity because of the fact that any adjective, hence a word suffixed with an participle, is allowed to become a noun with a zero derivation.

The list in Table 6 follows Göksel and Kerslake 2005, except the suffixes listed by them as converbial suffixes that require a postposition. The postposition in these cases will signal the adverbial function of postpositional phrase.

Most of these suffixes attach to an untensed verb. Except, the suffix -(y)ken which behaves much like the copular suffixes discussed above. Furthermore, the -(y)A in its subordinating function is typically used together with reduplication, e.g., *koşa koşa* 'run-(y)A run-(y)A = hurriedly', but also occurs in words like *diye*, where it does not need reduplication.[8]

Table 6: Subordinating suffixes and tags used for subordinating suffixes.

Tag	Suffix
⟨vn:inf⟩	-mA
⟨vn:inf⟩	-mAK
⟨vn:yis⟩	-(y)Iş
⟨vn:past⟩	-DIk
⟨vn:fut⟩	-(y)AcAk
⟨vn:res⟩	-(y)An
⟨part:past⟩	-DIk
⟨part:fut⟩	-(y)AcAk
⟨part:pres⟩	-(y)An
⟨cv:ip⟩	-(y)Ip
⟨cv:meksizin⟩	-mAksIzIn
⟨cv:ince⟩	-(y)IncA
⟨cv:erek⟩	-(y)ArAk
⟨cv:eli⟩	-(y)AlI
⟨cv:dikce⟩	-DIkCA
⟨cv:esiye⟩	-(y)AsIyA
⟨cv:den⟩	-dAn
⟨cv:den⟩	-zdAn
⟨cv:cesine⟩	-CAsInA
⟨cv:ya⟩	-(y)A
⟨cv:ken⟩	-(y)ken

Besides, some of the TAM markers, namely ⟨aor⟩, ⟨evid⟩, ⟨fut⟩, ⟨makta⟩ and less commonly ⟨cont⟩, derive adjectivals resembling relative clauses. TRmorph handles this by analyzing any verb with one of these TAM markers without further suffixes (e.g., agreement markers) as an adjective. For example, the word *okunmuş* in *okunmuş kitap* 'red book = the book that was read' is analyzed as 'oku⟨V⟩⟨pass⟩⟨V⟩⟨evid⟩⟨Adj⟩'.

[8] One may also analyze *diye* as a postposition, as it's use as subordinator is semantically unlike the others uses of -(y)A.

3.15 Productive derivational morphemes

Almost all the tags and relevant morphological process above are described as part of inflectional morphology in most grammar books. The suffixes described here are the ones that are traditionally considered derivational suffixes. Some of these suffixes, for example -lI and -sIz discussed earlier, may attach to word forms that are already inflected by other suffixes. Others normally attach only to the stem and produce another stem.

Of these suffixes, the noun–verb derivation suffix -lA causes a large number of ambiguous analyses since it is part of many other suffixes. These, for example, include the plural suffix -lAr whose remainder -r is also match with a verbal suffix (aorist). Hence, including -lA in the analysis causes an increase in the analyses of any plural noun. Currently, TRmorph analyzes -lA only after onomatopoeia. The rest of the verbs derived from nouns using this suffix are lexically specified.

Besides these sources of possible erroneous over-analyses, the derivational morphology specification in TRmorph over-generates in some cases. In particular, any form of the diminutive suffix is allowed to attach to any noun. The nouns typically do select for which suffix to use. However, there is no clear-cut way of predicting this. As a result, if used for generation, TRmorph will generate all diminutive suffixes for any noun, most likely only one of them being correct.

Furthermore, TRmorph does not limit the number of derivational suffixes that can be stringed one after another other. In reality this is a lot more restricted.

Table 7: Derivational morphemes analyzed by TRmorph. The column 'Derivation' lists the POS changes using two letter codes the first letter is the original POS, and the second one is the POS after the suffixation. Here, N means noun, J means adjective, A means adverb, V means verb and O is onomatopoeia.

Tag	Suffix	Derivation
⟨li⟩	-lI	NA NJ
⟨siz⟩	-sIz	NA NJ
⟨lik⟩	-lIk	NN JN AN
⟨dim⟩	-CIk	NN
	-cAk	
	-(I)cAk	
	-cAğIz	
⟨ci⟩	-CI	NN NJ
⟨ca⟩	-CA	NA AA JJ MJ
⟨yici⟩	-(y)IcI	VJ
⟨cil⟩	-CIl	NJ
⟨gil⟩	-gil	NN
⟨lan⟩	-lAn	JV
⟨las⟩	-lAş	NV JV
⟨yis⟩	-yIş	VN
⟨esi⟩	-(y)AsI	VJ
⟨sal⟩	-sAl	VJ
⟨la⟩	-lA	NV OV

4 About ambiguity and overgeneration

TRmorph produces more ambiguous analyses than the others (mainly based on Oflazer 1994) reported in the literature. This is partially because of the

5 Other tools

5.1 Stemming and lemmatization

5.2 Unknown word guesser

5.3 Hyphenation

5.4 Morphological segmentation

References

Beesley, Kenneth R and Lauri Karttunen (2003). "Finite-state morphology: Xerox tools and techniques". In: *CSLI, Stanford*.

Çöltekin, Çağrı (2010). “A Freely Available Morphological Analyzer for Turkish”. In: *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC2010)*. Valetta, Malta.

Göksel, Aslı and Celia Kerslake (2005). *Turkish: A Comprehensive Grammar*. London: Routledge.

Hulden, Mans (2009). “Foma: a finite-state compiler and library”. In: *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics: Demonstrations Session*. Association for Computational Linguistics, pp. 29–32.

Lindén, Krister, Miikka Silfverberg, and Tommi Pirinen (2009). “HFST Tools for Morphology–An Efficient Open-Source Package for Construction of Morphological Analyzers”. In: *State of the Art in Computational Morphology*. Ed. by Cerstin Mahlow and Michael Piotrowski. Communications in Computer andInformationScience. Springer, pp. 28–47. ISBN: 978-3-642-04130-3.

Oflazer, Kemal (1994). “Two-level description of Turkish morphology”. In: *Literary and Linguistic Computing* 9 (2).

Oflazer, Kemal, Dilek Hakkani-Tür, and Gokhan Tur (1999). “Design for a Turkish Treebank”. In: *Proceedings of the Workshop on Linguistically Interpreted Corpora, at EACL’99*. Bergen, Norway.

Oflazer, Kemal, Bilge Say, Dilek Zeynep Hakkani-Tür, and Gökhan Tür (2003). “Building a Turkish treebank”. In: Text, Speech and Language Technology. Springer, pp. 261–277. ISBN: 9781402013348.

Schmid, Helmut (2005). “A programming language for finite state transducers”. In: *Proceedings of the 5th International Workshop on Finite State Methods in Natural Language Processing (FSMNLP 2005)*. Helsinki, pp. 308–309.

Index