Word Sketches for Turkish

Bharat Ram Ambati, Siva Reddy, Adam Kilgarriff

Lexical Computing Ltd, UK
bharat.ambati@gmail.com, siva@sketchengine.co.uk, adam@lexmasterclass.com

Abstract

In this paper we present an approach for generating word sketches for Turkish language. Unlike the previous approaches of using manually crafted rules by an expert, we use an existing dependency parser to generate word sketches. We describe the process of collecting a 42 million word corpus, its parsing and generation of word sketches. We evaluate the word sketches in comparision with a shallow parser based word sketches on an external evaluation task called topic coherence.

Keywords: Word Sketches, Turkish, Sketch Grammar

1. Introduction

Word sketches are one-page, automatic, corpus-based summaries of a word's grammatical and collocational behaviour. They were first used in the production of the Macmillan English Dictionary (Rundell, 2002). At that point, word sketches only existed for English. Today, they are built into the Sketch Engine (Kilgarriff et al., 2004), a corpus tool which takes as input a corpus of any language and generates word sketches for the words of that language. It also automatically generates a thesaurus and 'sketch differences', which specify similarities and differences between near-synonyms.

Turkish is the 21^{st} largest language in the world, with over 50m speakers[1], yet until recently there were few language resources available for it (Oflazer, 1994). The last decade has seen much increased activity with new tools such as a morphological analyzer and disambiguator (Yuret and Ture, 2006) and dependency parser (Eryiğit et al., 2008).

We first gathered the corpus from the web using the 'Corpus Factory' as described in (Kilgarriff et al., 2010b), then cleaned and deduplicated it using the jusText and Onion tools (Pomikálek, 2011), then lemmatized and POS-tagged it with Yuret and Ture's tool. Up until now, the next step would have been to load it into the Sketch Engine, and to prepare a 'Sketch Grammar' which would be used for finite-state shallow parsing to identify grammatical relations. However for Turkish we did not have an expert available to write that grammar: what was available was a full parser (which we would also expect to be more accurate than a quickly-written finite state grammar). So, instead, we extended the Sketch Engine input formalism so that it could accept parser output in CONLL format[2]. Then we generate word sketches directly from the parser output. Here we present these first word sketches for Turkish, which are also the first word sketches to be the product of a full parser.

The paper is arranged as follows. We first describe the process of corpus collection in Section 2. Section 3 describes the complexities of Turksih language and extraction of parsers using a dependency parser. In section 4, we explain in detail about the process of the extraction of word sketches from parsed sentences. We conclude the paper with the evalution of these automatically generated Turkish word sketches.

2. TurkishWaC: A corpus of size 42 million words

The corpus is collected using Corpus Factory method (Kilgarriff et al., 2010b) for collecting large web corpora of many languages by piggybacking on the work done by any commercial search engine like Bing[3]. Several thousands of target language search queries are generated from Wikipedia and the corresponding hit pages are downloaded. The pages are filtered using a language model, and body text extraction, deduplication and encoding normalization are performed thus building a clean corpus. We replaced body-text extraction and deduplication tools with the current current state-of-art tools jusText and Onion respectively (Pomikálek, 2011).

The final corpus, TurkishWaC, is of size 42.2 million words, accessible from the Sketch Engine[4].

3. TurkishWaC Annotation

In this section, we first describe the linguistic complexities/properties of Turkish language. Then we describe different tools used to

3.1. Turkish Langauge

Turkish is an agglutinative language with rich morphology. Turkish words may be formed through very productive derivations, and can give rise to several inflected forms. As a result, number of possible word forms that can be generated from a root word increases substantially. A single word can have up to four or five derivations. Morphological structure of Turkish is represented by splitting words

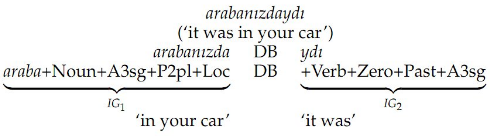
[1] `http://www.ethnologue.com` (accessed October 2011)

[2] `http://ilk.uvt.nl/conll/`

[3] `http://bing.com`

[4] `http://sketchenigne.co.uk`

into inflectional groups (IGs) The root and derivational elements of a word are represented by different IGs, separated from each other by derivational boundaries (DB). Each IG will have its own part of speech and inflectional features. An example taken from (Eryiğit et al., 2008) is shown below.



Turkish is a flexible constituent order langauge. Though the predominant order is SOV, as per requirements of the discourse context, constituents can freely change their position. It has been suggested that free-word order languages can be handled better using the dependency based framework than the constituency based one (**?**, **?**). There were efforts towards building a dependency parser for Turkish by Eryiğit et al. (2008). We employed the approach followed by Eryiğit et al. (2008) for parsing TurkishWaC corpus.

3.2. Parsing

Tools described in 22008Eryiğit et al.) require input corpus to be in latin-5 encoding. As a first step, we converted TurkishWaC which is in utf-8 encoding to latin-5. This data is processed with a two-level morphological analyzer of Oflazer (1994) to analyze each word. This morphological analyzer simultaneously produces derivational boundary (DB) and inflectional groups (IGs) which encode relevant features of a given word form. Turksih being a morphologically rich language, we need a morphological analyzer which accounts for this rich morphology. This morph analyzer Oflazer (1994) gives different possible morphological analyses for each word which includes POS tags as well. Out of these multiple analyses, we need to pick the correct analysis for each word. We used the morphological disambiguator of Yuret and Ture (2006) which has an accuracy of 96For the words not recognized by the morphological analyzer, we first checked if it is either a punctuation or a number and assigned the corresponding part-Of-speech (POS) tag. For the rest, we tagged them with proper noun as the POS tag.

Eryiğit et al. (2008) used MaltParser (Nivre and Hall, 2005) trained on Turkish dependency treebank data for parsing Turkish. MaltParser is a system for data-driven dependency parsing, which can be used to induce a parsing model from treebank data and to parse new data using an induced model. We selected Nivre Arc-Standard algorithm of MaltParser as it gave the best accuracy for Turkish language. Eryiğit et al. (2008) showed that using IGs as the basic parsing units rather than words improved parser performance. So, we used IGs as basic parsing units.

On a quadcore system, it took 10 days to parse the whole TurkishWaC. The final output is converted to utf-8 encoding for the post-processing tools that develop word scketches based on the parser output.

Sentence

We/PRP created/VB the/DET first/ADJ word/NN sketches/NN for/PREP Turkish/NN

Sketch Grammar

OBJECT:
1:[tag="VB"] [tag="DET"]{0,1} [tag="ADJ"] × [tag="NN"] *2:[tag="NN"]*

Figure 2: Sketch Grammar for OBJECT relation

4. Word Sketches from TurkishWaC

Figure 1 displays a sample output of Turkish parser (described above) in CONLL format. We wanted to generate word sketches from the parser's output.

The first step in generating word sketches is to generate dependency tuples. To date, Sketch Engine generates these tuples from a corpus using Sketch Grammar. For example, take the sentence and the sketch grammar displayed in Figure 2. The grammar rule means that the word with tag VB is in relation OBJECT with the word with tag NN, if VB is followed by an optional DET tag followed by any number of ADJs and NNs. This grammar rule generates the dependency tuple *(sketches, OBJECT, created)*, which means that *sketches* is the *OBJECT* of *created*.

Turkish language is morphologically very rich and is hard to write a grammar capturing all the linguistic phenomena. It has been shown by Eryiğit et al. (2008) that morphological features play a crucial role in identifying the dependency relations apart from POS tags. Instead Eryiğit et al. (2008) developed a dependency parser which considers into many linguistic criteria. A sample output of the parser is shown in figure **??**. The dependency information is represented in CONLL format.

We use the parser's output to generate dependency tuples from TurkishWaC without using Sketch Grammar. The column HEAD denotes that the current word is in relation DEPREL with the word whose column ID is equal to HEAD. For example, the lemma *ilgi* (ID:7) is the SUBJECT (column DEPREL) of the lemma *var* (ID:8). All the tuples generated from the sentence in Figure 1 are displayed in Figure 3. Apart from these, we also generate additional tuples depending upon the type of relation like symmetric (e.g. COORDINATION), dual (e.g. OBJECT/OBJECT_OF), unary (e.g. INTRANSITIVE), trinary (e.g. PP_IN) [more details in full paper].

(ki, INTENSIFIER, eğer), (ülkelere, OBJECT, ve),
(ve, COORDINATION, çek), (o, SUBJECT, var),
(özellik, DATIVE.ADJUNCT, var),
(ilgi, SUBJECT, var), (var, MODIFIER, çek),
(bu, DETERMINER, bölüm), (bölüm, SUBJECT, çek),
(ilgi, OBJECT, çek)

Figure 3: Dependency tuples from Figure 1

```
          ID      WORD            LEMMA      POSTAG  HEAD    DEPREL
=====================================================================
          1       Eğer            eğer       Conj    13      S.MODIFIER
          2       ki              ki         Conj    1       INTENSIFIER
          3       ülkelere        ülke       Noun    4       OBJECT
          4       ve              ve         Conj    12      COORDINATION
          5       onların         o-p        Pron    8       SUBJECT
          6       özelliklerine   özellik    Noun    8       DATIVE.ADJUNCT
          7       ilginiz         ilgi       Noun    8       SUBJECT
          8       varsa           var        Verb    12      MODIFIER
          9       bu              bu         Det     10      DETERMINER
          10      bölüm           bölüm      Noun    12      SUBJECT
          11      ilginizi        ilgi       Noun    12      OBJECT
          12      çekebilir       çek        Verb    13      SENTENCE
          13      .               .          Punc    0       ROOT
```

Figure 1: A sample output of the parser in CONLL format

Once these tuples are generated, to create a word sketch for a target word, we rank all its collocations (words in relation with the target word) in each grammatical relation using logDice (Curran, 2004). [More details about statistical measures later].

Word Sketch of the words *ekmek (bread)* and *çay (tea)* for selected grammatical relations are displayed in Figure 4 and Figure 5.

5. Thesaurus from Word Sketches

Using these word sketches, we have also built a distributional thesaurus for Turkish.

6. Evaluation

The typical evaluation of word sketches is performed manually by lexicographers who are native speakers of the target language. Hundreds of words are chosen for evaluation, and word sketches for these words are evaluted by lexicographers by grading (word, relation, collocation) triplets for their ability to form a meaninginful entry in collocations dictionary (Kilgarriff et al., 2010a). Higher the average score over all the triplers, higher is the accuracy of word sketches. However in the case of Turkish, we could not hire enough expert lexicographers. Additionally, manual evaluation of word sketches is time-taking and expensive, which we hope to do in future if we could find interested parties.

Instead, we opted for an automatic evaluation of word sketches. We evaluated word sketches on the task of topic coherence. A topic is a bag of words which are similar to each other and describe a coherent theme. In the task of topic coherence, given a topic, we score the topic for its coherence. Higher the similarity between words in the topic, higher is the coherence. To find the similarity between two words, we make use of thesaurus generated from the word sketches (described in Section **??**). Our intuition is that for a given coherent topic, the topic coherence score predicted by a thesaurus generated from high quality word sketches is higher than the score by a thesuarus generated from low quality word sketches.

6.1. Coherent Topic Selection

We use Turkish WordNet to choose coherent topics. A wordnet synset (a synonym set) represents a highly coherent topic since all the words in the synset describe an identical meaning (topic). In the WordNet, synsets are arranged in hierarchy in which a synset is linked with its hypernyms, hyponyms, antonyms, meronyms, holynyms etc. A synset along with its linked synsets at a distance of one or two, represent a slightly coherent topic.
For example, figure **??** represent various topics. A topic built from a synset $\mathtt{S}$ and its related synsets at a distance $\mathtt{d}$ can be formally represented as a set of words $\mathtt{T} = \{\mathtt{w_i} : \mathtt{w_i} \in \mathtt{S}*\}$, where S* represents the union of the synset S and its related synsets. $\mathtt{S}* = \cup\, \mathtt{S_i}$ for all $\mathtt{S_i}$ s.t. $\mathtt{distance}(\mathtt{S},\mathtt{S_i}) <= \mathtt{d}$

6.2. Topic Coherence Score

For a given topic $\mathtt{T} = \{\mathtt{w_1}, \mathtt{w_2}, \ldots \mathtt{w_n}\}$, we calculate its coherence by the taking the average similarity over all the pairs of words in T.

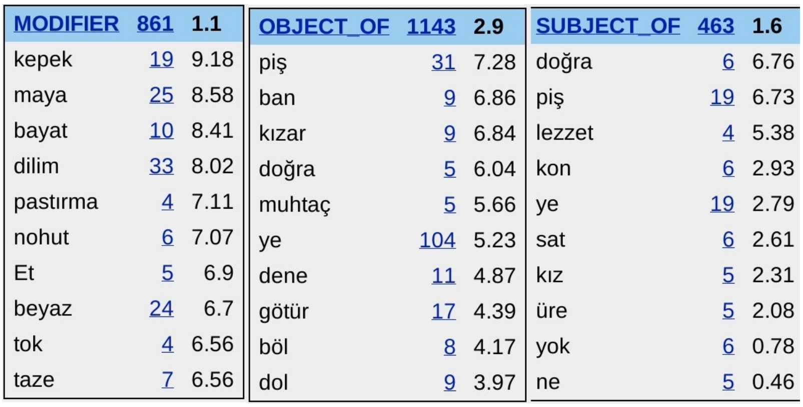
$$C_T = \frac{\sum\limits_{\mathtt{i,j}} \mathtt{sim}(\mathtt{w_i}, \mathtt{w_j})}{\mathtt{n} * (\mathtt{n} - 1)/2}$$

where $\mathtt{sim}(\mathtt{w_i}, \mathtt{w_j})$ represents the thesaurus similarity between the words $\mathtt{w_i}$ and $\mathtt{w_j}$. Over all the topics, we compute the average coherence score.

7. Results

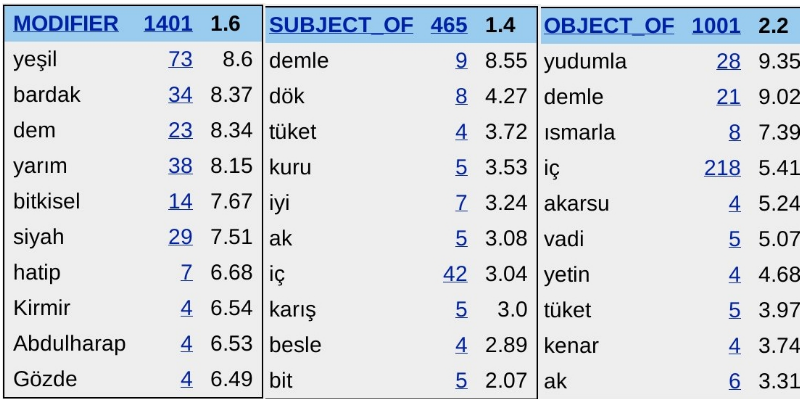
8. Summary

We collected and cleaned a corpus for Turkish. We identified leading NLP tools for Turkish and applied them to the corpus. We loaded the corpus into the Sketch Engine and developed a new module that allows us to prepare word sketches directly from CONLL-format output. We now have the first Turkish word sketches.
We intend to evaluate them (see Kilgarriff et al. (2010a)) before LREC in Turkey. We anticipate that they will be of interest to linguists, lexicographers, translators, and others working closely with, or studying, the Turkish language. These word sketches are currently available in Sketch Engine.



MODIFIER	861	1.1	OBJECT_OF	1143	2.9	SUBJECT_OF	463	1.6
kepek	19	9.18	piş	31	7.28	doğra	6	6.76
maya	25	8.58	ban	9	6.86	piş	19	6.73
bayat	10	8.41	kızar	9	6.84	lezzet	4	5.38
dilim	33	8.02	doğra	5	6.04	kon	6	2.93
pastırma	4	7.11	muhtaç	5	5.66	ye	19	2.79
nohut	6	7.07	ye	104	5.23	sat	6	2.61
Et	5	6.9	dene	11	4.87	kız	5	2.31
beyaz	24	6.7	götür	17	4.39	üre	5	2.08
tok	4	6.56	böl	8	4.17	yok	6	0.78
taze	7	6.56	dol	9	3.97	ne	5	0.46

Figure 4: Word Sketch of *ekmek (bread)*



MODIFIER	1401	1.6	SUBJECT_OF	465	1.4	OBJECT_OF	1001	2.2
yeşil	73	8.6	demle	9	8.55	yudumla	28	9.35
bardak	34	8.37	dök	8	4.27	demle	21	9.02
dem	23	8.34	tüket	4	3.72	ısmarla	8	7.39
yarım	38	8.15	kuru	5	3.53	iç	218	5.41
bitkisel	14	7.67	iyi	7	3.24	akarsu	4	5.24
siyah	29	7.51	ak	5	3.08	vadi	5	5.07
hatip	7	6.68	iç	42	3.04	yetin	4	4.68
Kirmir	4	6.54	karış	5	3.0	tüket	5	3.97
Abdulharap	4	6.53	besle	4	2.89	kenar	4	3.74
Gözde	4	6.49	bit	5	2.07	ak	6	3.31

Figure 5: Word Sketch of *çay (tea)*

References

Curran, J. (2004). *From Distributional to Semantic Similarity*. PhD thesis, Institute for Communicating and Collaborative Systems, School of Informatics, University of Edinburgh.

Eryiğit, G., Nivre, J., and Oflazer, K. (2008). Dependency parsing of turkish. *Computational Linguistics*, 34:357–389.

Kilgarriff, A., Kovar, V., Krek, S., Srdanovic, I., and Tiberius, C. (2010a). A quantitative evaluation of word sketches. In *Proceedings of the XIV Euralex International Congress*, Leeuwarden : Fryske Academy.

Kilgarriff, A., Reddy, S., Pomikálek, J., and PVS, A. (2010b). A corpus factory for many languages. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta.

Kilgarriff, A., Rychly, P., Smrz, P., and Tugwell, D. (2004). The Sketch Engine. In *Proceedings of EURALEX*, pages 105–116, Lorient, France, July.

Nivre, J. and Hall, J. (2005). Maltparser: A language-independent system for data-driven dependency parsing. In *In Proc. of the Fourth Workshop on Treebanks and Linguistic Theories*, pages 13–95.

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

Pomikálek, J. (2011). *Removing Boilerplate and Duplicate Content from Web Corpora*. PhD thesis, Masaryk University.

Rundell, M. (2002). *Macmillan English Dictionary for Advanced Learners*. Macmillan Education.

Yuret, D. and Ture, F. (2006). Learning morphological disambiguation rules for turkish. In *NAACL*, pages 328–334.