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

Word sketches are one-page, automatic, corpus-based summaries of a word’s grammatical and collocational behaviour. In this paper we present word sketches for Turkish. Until now, word sketches have been generated using a purpose-built finite-state grammars. Here, we use an existing dependency parser. We describe the process of collecting a 42 million word corpus, parsing it, and generating word sketches from it. We evaluate the word sketches in comparison with word sketches from a language independent sketch grammar on an external evaluation task called topic coherence, using Turkish WordNet to derive an evaluation set of coherent topics.

Keywords: Word Sketches, Turkish, Sketch Grammar, Dependency Parsing, Topic Coherence

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\textsuperscript{st} largest language in the world, with over 50m speakers\textsuperscript{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 parser which we would also expect to be more accurate. So, instead, we extended the Sketch Engine input formalism so that it could accept parser output in CONLL format\textsuperscript{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 parser.

2. TurkishWaC: A Turkish web corpus of 42 million words

The corpus was collected using the Corpus Factory method (Kilgarriff et al., 2010b). First, we gather a list of ‘seed words’ of the language from its Wikipedia\textsuperscript{3}. Then we generate several thousand search engine queries by randomly selecting three seed words. We then send these queries to a commercial search engine (in this case, Bing\textsuperscript{4}). We then gather all the pages that Bing identifies in its hits pages. 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 state-of-art tools jusText and Onion respectively (Pomikálek, 2011).

The final corpus, TurkishWaC\textsuperscript{5}, is of size 42.2 million words and is accessible within the Sketch Engine\textsuperscript{6}.

3. TurkishWaC Annotation

In this section, we first describe some relevant linguistic properties of Turkish, and then we describe different tools used to process TurkishWaC.

Turkish is an agglutinative language with rich morphology. Turkish words may be formed through very productive processes, and may have many inflected forms. The morphological structure of a Turkish word may be represented by splitting the word 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.

\begin{tabular}{ll}
\texttt{arabanzdaya} & (‘it was in your car’) \\
\texttt{arabanza} & \texttt{gal} \\
\texttt{araba+Noun+A3sg+P2pl+Loc} & \texttt{+Verb+Zero+Past+A3sg} \\
$\kappa_1$ & $\kappa_2$ \\
‘in your car’ & ‘it was’
\end{tabular}

\textsuperscript{3}http://dumps.wikimedia.org/trwiki
\textsuperscript{4}http://bing.com
\textsuperscript{5}WaC stands for the acronym Web as Corpus.
\textsuperscript{6}http://sketchengine.co.uk

\textsuperscript{1}http://www.ethnologue.com (accessed October 2011)
\textsuperscript{2}http://ilk.uvt.nl/conll/
Turkish is a flexible constituent order language. Though the predominant order is SOV, constituents can freely change their position according to the requirements of the discourse context. It has been suggested that free-word order languages can be handled better using a dependency framework rather than a constituency-based one (Hudson, 1984; Shieber, 1985).

We needed a morphological analyzer which accounted for this rich morphology. Oflazer (1994) describes such an analyzer. It is a two-level analyzer which produces derivational boundary (DB) and inflectional groups (IGs). It gives different possible morphological analyses, including part-of-speech (POS) tags, for each word. We first converted from UTF-8 (the encoding in which TurkishWaC had been prepared) into Latin-5 (as required for the tools we were to use). We then applied Oflazer’s morphological analyzer to the corpus. Out of the multiple analyses that were output, we needed to select the contextually correct one for each word. We used the morphological disambiguator of Yuret and Ture (2006) which has an accuracy of 96% for this purpose. For a word not recognized by the morphological analyzer, we first checked if it was either a punctuation mark or a number and, if it was, assigned the corresponding POS tag. For the rest, we tagged them as proper nouns.

Eryiğit et al. (2008) used MaltParser (Nivre and Hall, 2005) trained on a 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.

Figure 1 displays a sample output of Turkish parser in CONLL format. On a quadcore system, it took 10 days to parse the whole TurkishWaC.

4. Word Sketches from TurkishWaC

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.

4.1. Word Sketches using Turkish dependency parser

Since Turkish had an existing parser which provides dependency information, we aim to make use of parser’s output rather than writing a sketch grammar to generate dependency tuples. In figure 1, 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_IN), unary (e.g. IN-TRANSITIVE), trinary (e.g. PP_IN).

Once these tuples are generated, we rank all its collocations (words in relation with the target word) in each grammatical relation using logDice (Curran, 2004; Rychlý, 2008) and create a word sketch for a target word.

The word sketches of the word *ekmek* (bread) for selected grammatical relations are displayed in Figure 4.

### Universal Sketch Grammar

Recently, we designed a sketch grammar which can be applied for any corpora irrespective of the language, and so is the name Universal Sketch Grammar. The grammar aims to capture word associations of a given word. We define relation names based on the location of the context words w.r.t. the target word. For example, all the verbs located left to a word within a distance of three from the target word are in the relation *verb_left* with the target word. The grammar describing this rule is

```plaintext
=verb_left
2:[tag="V.*"] [tag=".*"]{0,3} 1:[]
```

Similarly we define the relations *verb_right*, *noun_left*, *noun_right*, *adjective_left*, *adjective_right*, *adverb_left* and *adverb_right*. Additionally we define the relations *nextleft* and *nextright* for the words immediately next to a given word. We also capture conjunction using the following rule.

```plaintext
=conj
1:[ ] [tag="C.*"] 2:[ ]
```

Figure 5 display the word sketches from universal sketch grammar.

### 5. Thesaurus from Word Sketches

In Sketch Engine, distributional thesaurus can be built for any language if the word sketches of the language exist. The thesaurus is built by computing similarity between words based on the extent of overlap between their word sketches. In contrast to earlier approaches of building a distributional thesaurus (Lin, 1998), Sketch Engine’s implementation (Rychlý and Kilgarriff, 2007) is known for its speed with most thesauri computation taking less than an hour. The thesaurus can also cluster similar words into different groups which share common meaning. Since word sketches for Turkish exist, we have also built its distributional thesaurus. Figures 6 and 7 display the distributional thesaurus entries of the word *ekmek* (bread) from dependency parser and universal sketch grammar.
6. Evaluation

The typical evaluation of word sketches is performed manually by lexicographers who are native speakers of the target language. A sample of words is chosen for evaluation, and word sketches for these words are evaluated by lexicographers who assess, for each collocation, whether they would include it in a published collocations dictionary (Kilgarriff et al., 2010a). The higher the average score over all the collocations, the higher is the accuracy of the word sketches.

However in the case of Turkish, we did not have access to lexicographers. Instead, we opted for an automatic evaluation of word sketches. Reddy et al. (2011) used word sketches in an external task called semantic composition. Inspired from it, we evaluate word sketches on an another external task, the task of topic coherence (Newman et al., 2010). 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. The higher the similarity between words in the topic, the higher is the coherence. To find the similarity between two words, we make use of thesauri generated from word sketches. 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 from a thesaurus generated from low quality word sketches.

<table>
<thead>
<tr>
<th>Lemma</th>
<th>Score</th>
<th>Freq</th>
</tr>
</thead>
<tbody>
<tr>
<td>yemek</td>
<td>0.198</td>
<td>8923</td>
</tr>
<tr>
<td>yiyecik</td>
<td>0.17</td>
<td>1559</td>
</tr>
<tr>
<td>meyve</td>
<td>0.157</td>
<td>4279</td>
</tr>
<tr>
<td>süt</td>
<td>0.15</td>
<td>5093</td>
</tr>
<tr>
<td>yumurta</td>
<td>0.149</td>
<td>3052</td>
</tr>
<tr>
<td>kahve</td>
<td>0.139</td>
<td>2027</td>
</tr>
<tr>
<td>şarap</td>
<td>0.133</td>
<td>1472</td>
</tr>
<tr>
<td>sebzə</td>
<td>0.132</td>
<td>1693</td>
</tr>
<tr>
<td>balık</td>
<td>0.131</td>
<td>5524</td>
</tr>
<tr>
<td>hamur</td>
<td>0.129</td>
<td>1115</td>
</tr>
<tr>
<td>peynir</td>
<td>0.128</td>
<td>912</td>
</tr>
<tr>
<td>yoğurt</td>
<td>0.123</td>
<td>1253</td>
</tr>
</tbody>
</table>

Table 1: Topic coherence scores of thesauri over WordNet

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 WordNet, synsets are arranged in hierarchy in which a synset is linked with its hypernyms, hyponyms, antonyms, meronyms, holonyms etc. A synset along with its linked synsets at a distance of one or two also represent a topic, but with a different degree of coherence.

A topic built from a synset \( S \) and its related synsets at a distance \( d \) can be formally represented as a set of words \( T = \{ w_i : w_i \in S^* \} \), where \( S^* \) represents the union of the synset \( S \) and its related synsets. \( S^* = \bigcup S_i \) for all \( S_i \) s.t. \( \text{distance}(S, S_i) < = d \).

6.2. Topic Coherence Score

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

\[
C_T = \frac{\sum_{i,j} \text{sim}(w_i, w_j)}{n \times (n - 1)/2}
\]

where \( \text{sim}(w_i, w_j) \) represents the thesaurus similarity between the words \( w_i \) and \( w_j \).

7. Results

We compute the average topic coherence score over all the WordNet synsets using both the thesauri generated from dependency parser output and universal sketch grammar, and compare coherence scores of each other to evaluate word sketches. The higher the coherence, the better are the word sketches. Our assumption is that wordnet synsets are highly coherent. Table 1 displays the results of topic coherence over synsets at a distance of 0, 1 and 2.

From the results we observe that topic coherence of nouns and verbs at synset level is higher for thesaurus from dependency parser. This gives us an idea that word sketches of noun and verb from dependency output are more informative/accurate than from universal sketch grammar. As the distance increases, the coherence score of verbs is consistently higher for dependency parser based word sketches. This shows that dependency parser is good at capturing
verb’s properties. For nouns, it is unclear why the coherence score from dependency parser is lower than universal sketch grammar at a distance of one.

For adjectives, interestingly, universal sketch grammar perform better. In our analysis we found the reason perhaps could be due to conjunction. The dependency parser always mark the conjunct word as the word in relation with target word, e.g. in the phrase sari/yellow ve/and kırmızı/red, kırmızı is in relation conjunction with ve, resulting in the tuple (ve, conj, kırmızı) instead of (sarı, conj, kırmızı). The universal sketch grammar generates the latter tuple. A new grammatical rule which can generate the latter tuple can be written using trinary relations in Sketch Engine but we leave this work for future.

As the distance increases i.e. as the topic becomes generalized, the topic coherence is expected to decrease. But at some cases we find there is an increase in topic coherence. This might be due to fine grained classification of WordNet synsets.

Overall the results suggest that dependency parser based word sketches of nouns and verbs are relatively accurate and informative than universal sketch grammar. It is the opposite case for adjectives. We leave a thorough study on these differences for future when we have adequate resources.

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. In addition, we presented universal sketch grammar which is language independent grammar. We generated two different thesauri from these word sketches.

We evaluated dependency parser based word sketches with universal sketch grammar by evaluating them on an external task of evaluation, the topic coherence using Turkish WordNet synsets and the thesauri generated from word sketches. Our results show that both the dependency parser based sketches are more accurate for verbs and nouns than simple sketch grammar.

In the future, we aim to build word sketches from our recent large (more than a billion size) corpora of Turkish (Baisa and Suchomel, 2012) and other Turkic languages. We anticipate that word sketches and thesauri 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.

9. Acknowledgements

We would like to thank Gülşen Cebiroğlu Eryiğit and Kemal Oflazer for their kind help on providing Turkish tools. We would also like to thank the reviewers for their suggestions on improving this work.

References


