Precomputed Word Embeddings for 15+ Languages

Ondřej Herman[1,2]

[1] Faculty of Informatics
Masaryk University
Botanická 68, 612 00 Brno
Czech Republic
xherman1@fi.muni.cz
[2] Lexical Computing s.r.o.
Botanická 68, 612 00 Brno
Czech Republic
ondrej.herman@sketchengine.eu

Abstract. We calculated word embedding models using fastText for multiple languages and corpora. The models are available for download and through a Web interface at https://embeddings.sketchengine.eu/.

Keywords: Word embeddings · Sketch Engine · Corpora

1 Word Embeddings

Word embeddings serve as an useful resource for many downstream natural language processing tasks. The embeddings map or embed the lexicon of a language onto a vector space, in which various operations can be carried out easily using the established machinery of linear algebra. The unbounded nature of the language can be problematic and word embeddings provide a way of compressing the words into a manageable dense space.

The position of a word in the vector space is given by the context the word appears in, or, as the distributional hypothesis postulates, *a word is characterized by the company it keeps* [2]. As similar words appear in similar contexts, their positions will also be close to each other in the embedding vector space. Because of this many useful semantical properties of words are preserved in the embedding vector space.

2 Models

The models were created using a modified version of the fastText [1] package with the ability to read corpora as indexed by the Manatee corpus manager, which is the core of the Sketch Engine [4]. This allows us to calculate models to have identical tokenization and format as the source corpora.

The models are calculated with a dimension of **100**, which is reasonable trade-off between size and performance for common applications. The minimum frequency for the lexicon elements has been chosen to be **5**, as for tokens

A. Horák, P. Rychlý, A. Rambousek (eds.): Proceedings of Recent Advances in Slavonic Natural Language Processing, RASLAN 2021, pp. 41–46, 2021. © Tribun EU 2021

with fewer appearances it is rarely possible to estimate quality word vectors. The **skip-gram** model has been chosen for the calculation. It is slightly more expensive to evaluate compared to the continuous-bag-of-words model, but the vector quality for rare words is improved. The negative-sampling parameter has been reduced to 3, as for large corpora this has negligible influence on the performance of the resulting model, while the training speed is greatly improved.

2.1 Source Corpora

Most of the models are based on the TenTen family of corpora [3]. These corpora have been built from texts obtained from the Web. The texts contained in the corpora are cleaned and deduplicated, and where available, the text is also available in lemmatized form and with part-of-speech annotations. The corpora can be accessed from the Sketch Engine[3].

For most of the corpora, multiple models are available. There is always a base model calculated from the **word** attribute, which represents the raw corpus text. A **lc** model is calculated from a lowercased variant of the corpus. A **lemma** model uses the corpus with every word converted to their base forms. A **lemma_lc** model is a lowercased variant of the **lc** model. A **lempos** model combines lemmata with a part-of-speech annotations appended. The Table 1 shows a selection of the models available with the respective lexicon sizes.

Table 1: Model Lexicon Sizes

Corpus	**lc**	**lemma**	**lemma_lc**	**lempos**	**word**
Arabic					2197469
Czech		2386157	2147712		3900455
Danish		1854619	1854541	1930823	2722811
German		6917255	7147030	6576701	6996045
Early English	799595	907219	776060	990898	962268
English	5929132	5941733	5268157	6143073	6658558
English (BNC2)		145773	130468	153041	200565
Spanish	3200355	2938116	2928086	3108981	3840913
Estonian	2915876	1906368			3307785
French	3581976	3971686	3304428	4300514	4335469
Italian	1325186	1363078	1134964	1508063	1624666
Korean					2949340
Portuguese	1872044	1700285	1700285	1783936	2264516
Russian	7494969	7770940	7205918	7858430	8340643
Slovenian	1143192	780745			1365370
Chinese					1636645

[3] `https://www.sketchengine.eu`

2.2 Data Format

The models are available for download in two different formats. Models with the `bin` extension are encoded in the native binary fastText format, while models with the `vec` extension use the textual Word2Vec format. We recommend the `bin` format, as it contains the subword n-gram information, is more compact and also faster to load.

2.3 Licensing

The models are available under the terms of the *Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License*[4]. This means that you can use the models for any non-commercial purposes and create derivative works based on the models, but you must give us credit and the derivative work needs to be available under the same terms.

3 Embedding Viewer

We also make the models accessible through a Web interface, which is hosted at `https://embeddings.sketchengine.eu/`. All the models which are available for download can also be examined through this interface.

The interface supports multiple types of queries. When a single word is entered, the words closest to it, according to cosine similarity, are retrieved and sorted by decreasing similarity.

When multiple words are entered, their word vectors are averaged and the result set consists of the words closest to the average value.

When a word in the query is prefixed with a minus ('-') character, the *inverse* of its word vector will be used, enabling to carry out arithmetic on the word vectors. For example, to obtain the result of *king - man + woman*, as formulated in [5], the user shall enter the query `king -man woman`. The result can be seen in the Figure 1.

3.1 API

In addition to the human-readable interface, the models can also be queried in an automated way and the result can be provided in machine-readable way. The supported formats are JSON and TSV.

The endpoint at `https://embeddings.sketchengine.eu/` accepts the following parameters:

Providing at least one of the `q`, `pos` or `pos_vec` parameters is mandatory, other parameters are optional.

The parameters are identical to the ones generated by the HTML user interface, so a link copied from the browser provides a good starting point for further experiments. As an example, retrieving the top 5 most similar lemmata

[4] Avaliable at `https://creativecommons.org/licenses/by-nc-sa/4.0/`.

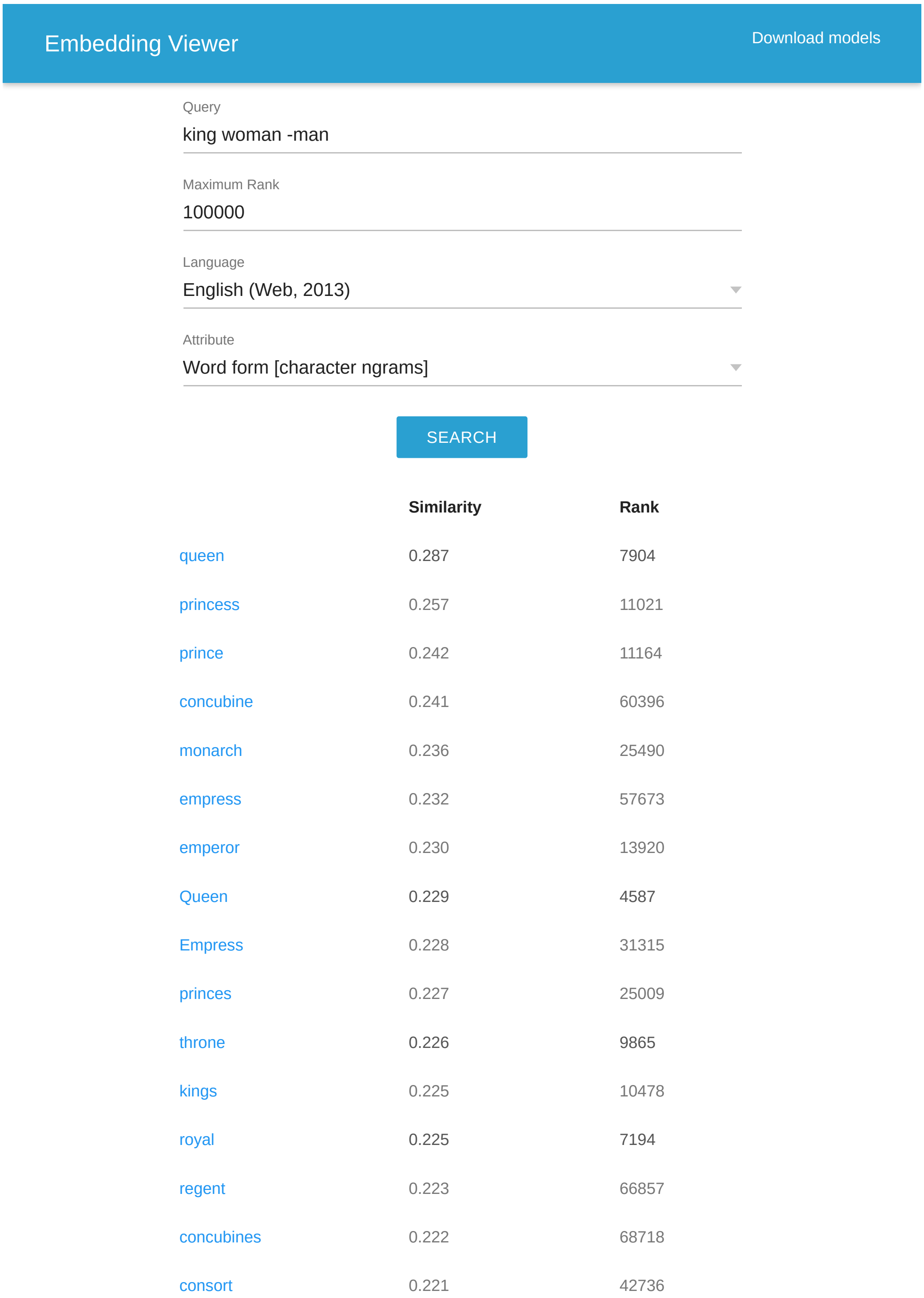

Fig. 1: Embedding Viewer

Table 2: Embedding API Query Parameters

Parameter	Description
q=QUERY	a complete query formatted as described above
pos=WORD	a single query word, can be specified multiple times
neg=WORD	a single query word complement, can be specified multiple times
pos_vec=VEC	same as pos, but interpreted as a comma-separated vector
neg_vec=VEC	same as neg, but interpreted as a comma-separated vector
n=N	the amount of rows to be returned
lim=N	maximum rank of the result entries
model=NAME	name of the embedding model
json	format the result as JSON
raw	format the result as TSV (tab-separated columnar format)
vec	include the word vectors in the result

to the lemma *dog* according to the English (Web, 2013) model in tab-separated format can be carried out by the 'curl' program[5].

```
$ curl 'https://embeddings.sketchengine.eu/?q=dog&lim=100000&n=5&
      model=English+%28Web%2C+2013%29%7CLemma&raw'

   puppy  0.8980982303619385 4139
   cat    0.8976492285728455 1678
   canine 0.8802799582481384 8694
   pup    0.8700659275054932 9166
   pet    0.8562509417533875 1622
```

Should you need lemmata similar to the lemma *cat* formatted as JSON, use the following query instead:

```
$ curl 'https://embeddings.sketchengine.eu/?q=cat&lim=100000&n=5&
      model=English+%28Web%2C+2013%29%7CLemma&json'

   {"w":[
       ["dog", 0.8976492881774902, 685],
       ["kitten", 0.8868610858917236, 8330],
       ["feline", 0.8669211864471436, 15259],
       ["pet", 0.8627837896347046, 1622],
       ["chinchilla", 0.8478652834892273, 51731]]
   }
```

The tab-separated format is easily usable for shell scripting and other similar "free-form" approaches, while JSON might be more appropriate for integration into more complex systems, in which the regular standardized form provides full control over the parsing details.

[5] Available from `https://curl.se/` for all common operating systems.

4 Future Work

The models which we have currently published cover only the most common languages. As we keep creating new corpora and extend existing ones, we will publish updated models in the future.

Of special interest might be models for other languages for which we have the data available. Eventually we plan to create word embedding models for every language present in the Sketch Engine. At the time of writing this article, this amounts to over 100 languages.

5 Conclusion

We calculated word embedding models using fastText for multiple languages and corpora. The models are available for download and through a Web interface at `https://embeddings.sketchengine.eu/`.

Acknowledgements. This work has been partly supported by the Ministry of Education of CR within the LINDAT-CLARIAH-CZ project LM2018101. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731015.

References

1. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics **5**, 135–146 (2017)
2. Harris, Z.S.: Distributional structure. Word **10**(2-3), 146–162 (1954)
3. Jakubíček, M., Kilgarriff, A., Kovář, V., Rychlý, P., Suchomel, V.: The tenten corpus family. In: 7th International Corpus Linguistics Conference CL. pp. 125–127 (2013)
4. Kilgarriff, A., Baisa, V., Bušta, J., Jakubíček, M., Kovář, V., Michelfeit, J., Rychlý, P., Suchomel, V.: The sketch engine: ten years on. Lexicography **1**(1), 7–36 (2014)
5. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. pp. 3111–3119 (2013)