From a dictionary towards the Hungarian Constructicon

Bálint Sass

Hungarian Research Centre for Linguistics, Institute for Lexicology
E-mail: sass.balint@nytud.hu

Abstract

We present the Hungarian Constructicon, a lexical resource which is an inventory of Hungarian constructions. It was derived mostly automatically from a dictionary. Main step of the processing was to identify constructions in the dictionary and lift them out and create individual entries for them. The tool is supplemented by a sophisticated online frontend which applies a so called dynamic toolbox to the constructicon database in order to be able to give an answer to any one-word or multiword query. Elements of this toolbox are analysed search, dynamic referencing and virtual entries which contains cross-references to elements of cxns present in the constructicon. In this way, the constructicon can handle inflected and derived forms in the query providing all plausible interpretations without needing to know a specific query formalism. This also covers the cases where a word can be interpreted as a regular form and an irregular form as well (cf. *'leaves'*). The Hungarian Constructicon combines the advantages of dictionaries and ccns and is equipped with an intuitive user interface.

Keywords: construction; constructicon; analysed search; dynamic referencing; virtual entry

1. Introduction

The term *constructicon* (Lyngfelt et al., 2018b: 97) (Fillmore, 2008: 49) (Jurafsky, 1991: 18) stands for the inventory of constructions of a language – by analogy to the term *lexicon*[1]. Accepting the position of Construction Grammar that utterances are not put together from words, but by combining cxns, it is quite straightforward that the basic unit of a lexical resource should be the cxn: a form–function pairing possibly spanning across linguistic levels. There is a considerable interest in developing (Lyngfelt et al., 2018a) and investigating (Dunn, 2023) ccns nowadays.

A ccn is not a list-like structure, but rather a network of cxns, employing different kinds of part-whole relations. Accordingly, a sophisticated cross-reference system is an important feature of ccns: from a cxn to its parts and vice versa. While in traditional dictionaries phrasemes, collocations and the like are often treated only incidentally (Fellbaum, 2016), ccns treat all kinds of meaning-bearing building units with equal care in a unified way as cxns, regardless of how complex they are. It is common to develop cnns by importing lexical information from existing lexical resources, especially from FrameNets.

In this paper, we present a Hungarian Constructicon (ccn-hu). The architecture of the ccn-hu consists of two components. The static part is the actual XML database of the

[1] Following Lyngfelt (2018) we will use the abbreviation *ccn* for constructicon and *cxn* (plural: *cxns*) for construction throughout this paper. We introduce the term *head-construction* (abbreviated as *hcxn*) denoting an entry in a ccn by analogy to the term *headword*. While cxn is a construction in general, hcxn is a concrete cxn having an entry in a ccn.

Hungarian cxns containing a structured entry for each cxn. The dynamic part is a set of tools which processes user input to find out which cxn (or cxns) are to be shown to the user and how. After discussing the status of single-element entities (Section 2), we compare the ccn-hu to other lexical resources (Section 3). Then, we elaborate on the static and the dynamic part in Section 4 and in Sections 5–7 respectively.

2. How many elements does a cxn contain?

An important point of our approach is that single-element units *are* cxns. Words are cxns, morphemes are cxns as well. Though this is not a novel idea (Goldberg, 2006: 5), we emphasize this here. In general, the cxn used to be a multi-element entity while the word (or the morpheme) is a single-element entity. Therefore, a common criterion for a cxn is to consist of at least two elements. Our idea is to include all units of meaning (Teubert, 2005) in the ccn regardless the number of their sub-elements. Namely, single-element entities are also treated as cxns, albeit a somewhat special case of them. The fundamental goal of ccns is to be able to handle all linguistic units in a unified way and allow the user to find all kinds of meaning-bearing units in them. By analogy, it is like to treat 1 (one) just like other positive integers, despite it is quite special being neither prime nor composite. This decision makes it easier to come up with simpler general descriptions of numbers or simpler methods which can handle all of them. We anticipate that the similar decision in lexicology, i.e. the inclusion of single-element entities into the notion of cxn, will have similar advantages.

Covering single-element entities, ccns can integrate dictionaries into themselves. In this way, a ccn can grasp the complete network of the language and show the connections between linguistic units.

3. Comparison to other lexical resources

In this section, we compare the Hungarian Constructicon to other ccns and then to some online dictionaries. As the ccn-hu contains single-element entities (e.g. simple words) as well (see Section 2), it can be considered a dictionary from a certain point of view, accordingly, the latter comparison is also relevant.

3.1 Comparison to ccns: Swedish and Russian

Many ccns have been created in recent years. We compare the approach and features of ours (ccn-hu) to the Swedish (Lyngfelt et al., 2018b) (ccn-sw) and to the Russian Constructicon (Janda et al., 2020; Bast et al., 2021) (ccn-ru) as well.

Size. Ccn-sw contains 393 cxns, ccn-ru contains 2277 cxns while ccn-hu contains more than 13000 cxns in the current version. It should be noted that the Hungarian cxns are less abstract for the most part. E.g. ccn-sw contains many high-level cxns like *'artikel'* ('article') or *'passiv'* ('passive'), but contains several more concrete cxns at the same time, like *'antingen X eller Y'* ('either X or Y') or *'dra OBJ'* ('drag OBJ') and non-mutable fixed ones like *'bla bla bla'* ('and so on'). The latter are also characteristic of ccn-hu.

Formalizedness. In our view, ccns should be formalized to the most possible extent rendering the database as machine readable as possible. We consider the level of formal-

ization to be low for both examined ccns. They seem to be just human readable, not inherently machine readable. This is future work for ccn-hu as well.

Connection to FrameNet. As the original English Constructicon (Fillmore, 2008) used FrameNet as starting point, it is somewhat surprising that none of the three ccns in question is directly connected to the corresponding FrameNet.

Single-element units. Differently from ccn-hu, neither ccn-sw nor cnn-ru contains single-element units.

Availability. Data of ccn-ru is freely available, it is in fact a simple `.csv` table. For ccn-sw, the whole cxn-list can be copied from the website. Concerning ccn-hu, the query interface will be freely usable for personal and research use, it is not decided yet whether the software and the data itself will be freely available or not.

3.2 Comparison to online dictionaries: DWDS and OALD

Here, we compare the ccn-hu to online dictionaries: DWDS (BBAW, 2023) and OALD (Oxford University Press, 2023) according to various aspects.

Multiword input. DWDS can not handle simple multiword input like *'das Buch'*. While *'zur Verfügung stellen'* is not handled, *'zur Verfügung haben'* is. OALD does not respond to *'black dog'*. Creating ccn-hu, one of our important aims is to be able to handle multiword input, even to give an answer to any possible query.

Irregular inflection. On the one hand, it is common that the irregular forms are included in dictionaries. DWDS is at least not totally complete in this sense as to the query *'Bücher'* it responds with the entry of *'der Bucher'* (booker) which is misleading. Similarly, *'hast'* takes the user to *'die Hast'*. However *'ziehst'* works well. These kind of redirecting is solved in ccn-hu by analysed search (see Section 5), cf. *'lovat'* (*'ló'* (horse) in accusative case).

On the other hand, it is also common that the regular forms are not included in dictionaries. Maybe they are considered out-of-scope and set aside as being part of "grammar". In OALD *'books'* silently redirects to *'book'*, it does not tell the user any information about the connection between the query and the resulting entry. For a language learner, this connection can be important. It holds especially for morphologically richer languages, it seems to be a good behaviour for a Finnish dictionary/ccn to respond with *'talo'* (house) + *'-ssa/-ssä'* (in) to the query *'talossa'*. This kind of redirecting is also solved in ccn-hu, cf. *'asztalt'* (*'asztal'* (table) in accusative case).

The most problematic case combines the above too: some word forms represent an irregular form of a word and a regular form of another word at the same time. In these cases dictionaries tend to present the irregular solution and tend to hide the regular, which can be misleading. Consider the English word form *'leaves'* and enter it to OALD. The irregular plural of *'leaf'* will be provided but the third person singular of *'leave'* does not. This solved in ccn-hu as well, cf. *'terem'*.

Use what you have. We take the position that it is better to have an incomplete entry for a cxn than nothing. DWDS responds to query *'Nagellackentferner'* with a partial entry

which contains cross-references to *'Nagellack'* and *'Entferner'* and some corpus examples, but no definition (cf. Janssen, 2008). This is clearly an automatically generated entry, but a very useful one: it helps the user understand the queried word. Virtual entries (see Section 7) implement this feature in ccn-hu.

Down-references. Cross-references from a cxn to its parts can be called *down-references*. DWDS do have down-references under *'Wortzerlegung'*word decomposition. OALD has it as well, it is accessible by double-clicking elements of cxns (see e.g. *'red herring'*). The ccn-hu has down references for every unit which has elements.

Formalizedness. Dictionaries are generally optimized for human-readability, so they tend to be less formalized compared to ccns. For example, DWDS still uses old-fashioned textual abbreviations like *'etw. jmd.'*.

We note that ccn-hu could be compared to a machine translation system too. The big picture is that such a systems usually work as a dictionary for one-word queries and as a translator for multiword queries, the ccn-hu works like a dictionary for multiword queries as well.

4. Lifting out cxns from a dictionary

In this section, we cover the static part of the ccn-hu, i.e. how we created its XML database.

In absence of a Hungarian FrameNet, we started from a monolingual dictionary and derive the ccn to a great extent automatically. Our initial dictionary was (Pusztai, 2003) which is a common reference work for Hungarian and contains more than 73000 entries. The automatic ccn-creation process was carried out as described in the following.

Firstly, we carried out some basic XML preprocessing: fixed UTF-8 character encoding, normalized whitespaces and lowercased the whole dictionary. Then we made the initial XML a bit more data-centric converting some text nodes to attribute nodes. For example, the homonymy indexes were converted from `<hom>1</hom>` to `<hom value="1"/>`. This was a simple *vertical operation*, i.e. a transformation which affects the dictionary as a whole at once.

After that we identified cxns in the "collocation" part of the dictionary entries (marked by the `<coll>` element in the initial XML), we lifted out the XML subtree representing the cxn and created a new individual entry for it on its own. The lemma of the new entry becomes the textual form of the cxn and part-of-speech is set to "cxn" simply. Then we created cross-references from the original place of the cxn to the newly created entry. (These links are colored green in the user interface, see Section 9.) Thus, an additional 14000 entries were added to the ccn being prepared.

An online lexical resource does not encounter any size limits, so we resolved common abbreviations and the tilde (∼) headword placeholder. The latter was not a trivial task as due to traditional practice in Hungarian lexicography in some cases the headword had to be altered before replacing the tilde.

In the final step we converted the ccn into a HTML form which is suitable for displaying and easily queryable using XSLT at the same time and added the entry–query links (see Section 8). Then, the finished material was put behind a Flask frontend for online use.

Goldberg (2006: 5) presents a long list of cxn types. Many different types appear in our final ccn database: bound morphemes, simple words, compound words, filled idioms. All these are non-mutable, continuous cxns. Handling more complex, mutable, non-continuous or partially filled cxns remains a future work (see Section 10).

5. Analysed search

Interacting with a ccn, you should have the opportunity to search for cxns not just words. We will introduce a new type of search called *analysed search* – suitable for ccns – to eliminate the need for users (e.g. language learners) to learn a formal language or a specific search tool (Sato, 2012). The user is allowed to enter free text in a plain search box, then we apply automatic morphological analysis to the text, and direct the user to the appropriate identified cxn(s). This process is applied to the ccn database described in Section 4 and performed for every type of cxn from simple or compound words to e.g. preverb-verb combinations.

Hungarian is a morphologically rich language (Megyesi, 1998) with an extensive inflectional and derivational system. Additionally, coumpounding is also happens inside the word, i.e. compounds are written together as one word. We use the `e-magyar` system (Indig et al., 2019) for processing user input. The `emMorph` (Novák et al., 2016) morphological analyser module can break down words into morphemes, for example *'gyerekeket'* is broken down to these elements: *'gyerek'* ('child') + *'-k'* ('plural') + *'-t'* ('accusative case'), or *'hatalmán'* to these: *'hatalom'* ('power') + *'-a'* ('possessive suffix') + *'-n'* ('on'). This is exactly what is needed because the basic elements of cxns are morphemes.

The algorithm of analysed search is the following for a one-word input:

1. if the input is a hcxn on its own, take it into account;
2. perform the morphological analysis;
3. consider all analyses and take one segmentation from each: choose the segmentation with the longest left side part which is a hcxn;
4. we omit possible duplications collecting results into a set.

If there are several alternative results at the end, all of them are considered and presented one after another.

There has been a long-standing debate about whether a certain Hungarian word is compound or not, what is the lemma (the base form) of a certain Hungarian word, i.e. which derivational suffix should be removed and which one should not. Our approach allows us to put this debate aside. Taking the ccn itself as an oracle, we say that if a compound or a derived form is present as a hcxn, then it is accepted as is. An example concerning compounds: *'rendőr'* ('order guard' = policeman) will be presented as a cxn as it is a hcxn, while *'kapuőr'* ('gate guard' = gatekeeper) will be presented as a compound of two words *'kapu'* (gate) and *'őr'* (guard). The 3rd point of the above algorithm implements this mode of operation.

We note that analysed search is one of the rare cases where a classic low-level natural language processing tool, i.e the morphological analyser, can be used not only for solving a subtle subtask but also directly to meet the needs of end users.

We also note that applying analysed search we make heavy use of the fact that the Hungarian Constructicon is an inherently online tool. It would be hard to include e.g. all compound words future users may ever think of in a printed dictionary.

6. Identification of cxns

In Section 5 we discussed the case of one-word input only. It is important that the input can be multiword naturally, in fact it can be any linguistic element or combination: a morpheme, a word, a phrase or even a short text. A major task is to be able to identify (possible multiword) cxns in multiword input. While a complete solution – handling e.g. complex non-continuous verbal cxns – remains future work (see Section 10), there is already a partial solution of this task handling two easier cases.

On the one hand, the system recognizes non-mutable continuous cxns on their own or even as a part of a query. The algorithm matches the input text greedily to the hcxns and gives the longest one as a result. For example, as *'ad hoc'* and *'ad hoc bizottság'* (ad hoc committee) are both hcxns the queries presented in Table 1 will provide the cxns shown.

	query	**identified cxns**
(a)	*'ad hoc dolog'*	*'ad hoc'* + *'dolog'* (thing)
(b)	*'ad hoc bizottság dönt'*	*'ad hoc bizottság'* (ad hoc committee) + *'dönt'* (decides)

Table 1: An illustration of the operation of the greedy cxn-identification algorithm. If there is a choice (see (b)), the longer cxn will be identified.

On the other hand, the system recognizes a kind of non-continuous cxns as well, namely the preverb-verb combinations. While the `emMorph` morphological analyser module (see Section 5) does all kinds of analyses inside tokens, `emPreverb` (Pethő et al., 2022) module adds the functionality of connecting separated preverb tokens to their verbs. In Hungarian the preverb (or verbal prefix) is written together with the verb in certain cases, but it constitutes an independent token in others, placed possibly several words away from the verb (cf. Megyesi, 1998: 9). The algorithm loops over the tokens of the input. Processing a verb, the algorithm picks up the corresponding preverb (if there is one), connects it to the verb and reanalyses the resulting connected form, and when it comes to a preverb which is already connected, the algorithm simply skips it. Table 2 shows an example of this feature using *'bejön'* (come in) in which *'be'* (in) is the preverb and *'jön'* (come) is the verb.

	query	**identified cxns**
(a)	*'bejön'*	*'bejön'* (come in)
(b)	*'most jön be'*	*'most'* (now) + *'bejön'* (come in)

Table 2: An illustration of preverb-verb cxn identification. The separated preverb in query (b) is handled properly.

7. Dynamic referencing and virtual entries

Analysed search (Section 5) is supplemented by a novel cross-referencing process called *dynamic referencing*. If the search query does not have a matching cxn, but its parts do, a so called *virtual entry* is created on-the-fly automatically: containing nothing but references to the parts. For example, *'almafa'* (apple tree) is a hcxn, so the user will get its entry immediately, but *'grépfrútfa'* (grapefruit tree) is not, so the virtual entry created will contain a link to *'grépfrút'* (grapefruit) and another to *'fa'* (tree) beyond the information that the original query is a compound construction.

Perhaps it is not surprising that an overwhelming majority of possible queries will result in a virtual entry. Let us review the following cases from the simplest to the most complex using different cxns, all containing the morpheme *'asztal'* (table).

1. **Simple word.** The query for a simple word, e.g. *'asztal'* will simply provide its original real entry from (Pusztai, 2003). Words that do have an inner structure but present in the ccn as a hcxn on their own will behave the same way, see e.g. *'asztalos'* ('table + *-s* suffix' = carpenter). Results for all the other query types below will be virtual entries.
2. **Suffixed word.** For example, *'asztalra'* ('table+onto' = onto table) not being a hcxn on its own, will be analysed and its two parts will be shown in a virtual entry as *'asztal'* (table) + *'-ra/-re'* (onto). Fortunately, case markers and other suffixes like *'-ra/-re'* have a real entry in the initial dictionary already, so they can be presented using a hand-crafted mapping between `emMorph` codes and them.
3. **Compound word.** Compounds are treated similarly as they are cxns containing more than one morphemes just like suffixed words. For example, *'faasztal'* (wooden table) will result in a virtual entry containing *'fa'* (wooden) + *'asztal'* (table).
4. **Sequence of words.** Word sequences are firstly tokenized using the `emToken` tokenizer module (Mittelholcz, 2017) and then treated according to point 2 token by token. The result will be a sequence of (virtual) entries, for example *'három'* (three) and *'asztal'* (table) for the query *'három asztal'* (three table).
5. **Non-mutable continuous cxn.** Fixed continuous cxns are identified inside query text (see Section 6), so *'nem az ő asztala'* ('not his table' = none of his business) will be found and its original entry will be presented.
6. **Non-continuous preverb-verb cxn.** These cxns are also identified (see Section 6), and will be presented as a real or a virtual entry.
7. Handling more complex cxns remains future work, see Section 10.

What if the meaning is more than the meaning of the parts presented in the virtual entry? This is a matter of completeness of the ccn. If a cxn is not present in the ccn, we can not do anything but show information about the parts of it. Obviously, in the present version of the Hungarian Constructicon we can only work with those cxns that were included in the initial dictionary.

We do not think that our ccn is complete in any sense, it just contains quite a large amount of cxns. Instead of trying to make the ccn complete at all costs, we focus on making it easy to expand. Clearly, any expansion will influence dynamic referencing as it will decrease the need for virtual entries. If a brand new entry for *'grépfrútfa'* (grapefruit

tree) will be added in the future to the ccn, its own real entry will be presented for this query and virtual entry creation will no longer be needed thenceforth. This behaviour was successfully tested in the system.

We can refer to analysed search, dynamic referencing and virtual entries together as *the dynamic toolbox*. The point of the dynamic toolbox is that it allows the ccn to give an answer *always* to *any* queries to the best of its ability. If the ccn itself improves, the responses will improve as well.

8. Entry–query links

Ccn-hu will also offer a feature called *entry–query* links, which adds to its overall convenience. This is a kind of cross-referencing system from a lexicographic perspective and a user-friendly feature from user experience point of view.

It means that every word in the text of real or virtual entries functions as a link to start a query that looks up the word itself in the ccn. Unsuprisingly, the entry for *'sárga'* (yellow) contains the word *'citrom'* (lemon) in the definition part. Just click on *citrom* to reach the entry of this very word. The whole dynamic toolbox machinery described in the previous sections will start working as if it would be a query entered directly by the user. This allows us to add entry–query links to every word appearing in the entries.

This feature can help investigating the ccn itself as a subject of lexicological research. We can examine lexical loops (cf. Levary et al., 2012), or the question whether members of the definition vocabulary are themselves defined (cf. Atkins & Rundell, 2008: 448).

9. Availability

The Hungarian Constructicon is available for the scientific community and the general public as well at `http://ccn.nytud.hu/intro`. Please authenticate (username: `eLex2023` password: `letssee`) and feel free to try all examples typesetted like *'példa'* presented in this paper.

The user interface consists of a simple search box and a short description of the system. There are some clickable examples in the description text. A small icon to the right of the *Search* button gives some information about what is going on in the background: ✓ means that the result is a real entry (cf. point 1 in Section 7); ✗ means that no result can be provided; and the magic wand which appears in other cases means that some elements of the dynamic toolbox was applied.

The implementation is based on python3, Flask, XML, lxml and XSLT technologies. Recognizing non-mutable continuous cxns (Section 6) uses a simple hash for finding cxns.

10. Future work

There are many directions in which our works can be further developed. Some of them are listed below from easy ones to difficult ones.

- Create *up-references*, i.e cross-reference every cxn from the entries of its elements (cf. *down-references* in Section 3.2).

- Test the Hungarian Constructicon with end users, collect and investigate real life queries, and shape further development along the learned lessons.
- Integrate other lexical resources which can be used as a cxn source (e.g. Sass & Pajzs, 2010).
- To support the tasks below, develop a formal representation of cxns, or use an existing one, if possible.
- Handle inflected form of multiword cxns. Can be considered as a special case of the next one.
- Handle complex non-continuous verbal cxns with or without free slots (cf. point 7 in Section 7). The difficulty of this task lies in the fact that elements (words and bound morphemes) of this kind of cxns can appear in several different order with possible intervening words. The representation is to be worked out as well as the algorithm which can efficiently use it. Dependency parsing may have a role in the solution.
- Refer to the appropriate meaning of any cxn and "grey out" the others on the user interface. Seems to be a very hard problem.

11. Summary

In this paper, we presented the current version of the Hungarian Constructicon (ccn), a lexical resource which is an inventory of Hungarian constructions (cxns). The ccn was derived mostly automatically from a dictionary. To be able to handle all kinds of linguistic units in a uniform way we included morphemes and words into the category of cxns. The main step of the processing was to identify cxns in the dictionary and lift them out creating individual entries for them. The number of entries was increased by about 20 percent in this way.

The ccn is supplemented by a sophisticated online frontend which applies a so called dynamic toolbox to the ccn database in order to be able to give an answer to any one-word or multiword query. Elements of this toolbox are analysed search which provides an analysed version of the input query, dynamic referencing which creates virtual entries containing cross-references to elements of cxns which are not present in the ccn.

In this way, the ccn can handle inflected and derived forms in the query providing all plausible interpretations without needing to know a specific query formalism. This also covers the cases where a word can be interpreted as a regular form and an irregular form as well like in case of the English example *'leaves'* or Hungarian example *'terem'*.

Combining the advantages of dictionaries and ccns we consider our methodology a step towards creating a general purpose "ultimate" lexical resource.

12. References

Atkins, B.T.S. & Rundell, M. (2008). *The Oxford Guide to Practical Lexicography*. Oxford University Press.

Bast, R., Endresen, A., Janda, L.A., Lund, M., Lyashevskaya, O., Mordashova, D., Nesset, T., Rakhilina, E., Tyers, F.M. & Zhukova, V. (2021). The Russian Constructicon. An electronic database of the Russian grammatical constructions. URL https://constructicon.github.io/russian.

BBAW (2023). DWDS – Digitales Wörterbuch der deutschen Sprache. Das Wortauskunftssystem zur deutschen Sprache in Geschichte und Gegenwart. URL https://www.dwds.de.

Dunn, J. (2023). Exploring the Constructicon: Linguistic Analysis of a Computational CxG. In *Proceedings of the Workshop on CxGs and NLP / SyntaxFest*. Association for Computational Linguistics.

Fellbaum, C. (2016). The Treatment of Multi-word Units in Lexicography. In P. Durkin (ed.) *The Oxford Handbook of Lexicography*. Oxford: Oxford University Press, pp. 411–424.

Fillmore, C.J. (2008). Border Conflicts: FrameNet Meets Construction Grammar. In E. Bernal & J. DeCesaris (eds.) *Proceedings of the XIII EURALEX International Congress*. Barcelona: Universitat Pompeu Fabra, pp. 49–68.

Goldberg, A.E. (2006). *Constructions at work: The nature of generalization in language*. Oxford University Press.

Indig, B., Sass, B., Simon, E., Mittelholcz, I., Vadász, N. & Makrai, M. (2019). One format to rule them all – The `emtsv` pipeline for Hungarian. In *Proceedings of the 13th Linguistic Annotation Workshop*. Florence, Italy: Association for Computational Linguistics, pp. 155–165. URL https://www.aclweb.org/anthology/W19-4018.

Janda, L., Endresen, A., Zhukova, V., Mordashova, D. & Rakhilina, E. (2020). How to build a constructicon in five years: The Russian example. *Belgian Journal of Linguistics*, 34, pp. 162–175.

Janssen, M. (2008). Meaningless Dictionaries. In E. Bernal & J. DeCesaris (eds.) *Proceedings of the XIII. EURALEX International Congress*. Institut Universitari de Lingüística Aplicada, Universitat Pompeu Fabra, Barcelona, pp. 409–420.

Jurafsky, D. (1991). *An On-line Computational Model of Human Sentence Interpretation: A Theory of the Representation and Use of Linguistic Knowledge*. Ph.D. thesis, Department of Electrical engineering and computer sciences, University of California, Berkeley.

Levary, D., Eckmann, J.P., Moses, E., & Tlusty, T. (2012). Loops and Self-Reference in the Construction of Dictionaries. *Phys. Rev.*, X(2), p. 031018.

Lyngfelt, B. (2018). Introduction: Constructicons and constructicography. In Lyngfelt et al. (2018a), pp. 1–18.

Lyngfelt, B., Borin, L., Ohara, K. & Torrent, T.T. (eds.) (2018a). *Constructicography: Constructicon development across languages*. Amsterdam: John Benjamins.

Lyngfelt, B., Bäckström, L., Borin, L., Ehrlemark, A. & Rydstedt, R. (2018b). Constructicography at work: Theory meets practice in the Swedish Constructicon. In Lyngfelt et al. (2018a), pp. 41–106.

Megyesi, B. (1998). The Hungarian Language: A Short Descriptive Grammar.

Mittelholcz, I. (2017). `emToken`: Unicode-képes tokenizáló magyar nyelvre. [`emToken`: a Unicode-capable tokenizer for Hungarian.]. In V. Vincze (ed.) *MSZNY2017*. Szegedi Tudományegyetem, Informatikai Tanszék csoport, pp. 70–78.

Novák, A., Siklósi, B. & Oravecz, Cs. (2016). A New Integrated Open-source Morphological Analyzer for Hungarian. In N. Calzolari, K. Choukri, T. Declerck, S. Goggi, M. Grobelnik, B. Maegaard, J. Mariani, H. Mazo, A. Moreno, J. Odijk & S. Piperidis (eds.) *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*. Paris, France: European Language Resources Association (ELRA).

Oxford University Press (2023). Oxford Advanced Learner's Dictionary. URL https://www.oxfordlearnersdictionaries.com.

Pethő, G., Sass, B., Kalivoda, Á., Simon, L. & Lipp, V. (2022). Igekötő-kapcsolás [Connecting perverbs to verbs]. In G. Berend, G. Gosztolya & V. Vincze (eds.) *MSZNY 2022*. Szegedi Tudományegyetem, Informatikai Intézet, pp. 77–91.

Pusztai, F. (ed.) (2003). *Magyar Értelmező Kéziszótár [Hungarian Monolingual Explanatory Dictionary]*. Akadémiai Kiadó.

Sass, B. & Pajzs, J. (2010). FDVC – Creating a Corpus-driven Frequency Dictionary of Verb Phrase Constructions for Hungarian. In S. Granger & M. Paquot (eds.) *Proceedings of eLex 2009*. Louvain-la-Neuve: Presses universitaires de Louvain, pp. 263–272.

Sato, H. (2012). A Search Tool for FrameNet Constructicon. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*. Istanbul, Turkey: European Language Resources Association (ELRA), pp. 1655–1658.

Teubert, W. (2005). My version of corpus linguistics. *International Journal of Corpus Linguistics*, 10(1), pp. 1–13.