Splitting and Identifying Czech Compounds: A Pilot Study

Emil Svoboda
Charles University
Faculty of Mathematics and Physics
Prague, Czech Republic
svoboda@ufal.mff.cuni.cz

Magda Ševčíková
Charles University
Faculty of Mathematics and Physics
Prague, Czech Republic
sevcikova@ufal.mff.cuni.cz

Abstract

We present pilot experiments on splitting and identifying Czech compound words. We created an algorithm measuring the linguistic similarity of two words based on finding the shortest path through a matrix of mutual estimated correspondences between two phonemic strings. Additionally, a neural compound-splitting tool (*Czech Compound Splitter*) was implemented by using the Marian Neural Machine Translator framework, which was trained on a data set containing 1,164 hand-annotated compounds and about 280,000 synthetically created compounds. In compound splitting, the first solution achieved an accuracy of 28% and the second solution achieved 54% on a separate validation data set. In compound identification, the *Czech Compound Splitter* achieved an accuracy of 91%.

1 Introduction

Compounding refers to "the formation of a new lexeme by adjoining two or more lexemes" (Bauer, 2003, p. 40). For many languages, including Sanskrit, English and German, the process has been mapped and modelled extensively in static data resources and procedural tools, but this is not the case for Czech.

The present paper focuses on compounding in Czech, which is a language where compounds are nearly always represented in writing as a single string of graphical symbols unbroken by whitespace (from here: *graphical word*). The problem we tackle is twofold: a) upon being given a graphical word, to decide whether or not it is a compound; and b) upon being given a confirmed compound, to return the citation forms of its base words (from here: *parent words* or *parents*). Task a) will be referred to as *compound identification* and is approached as an instance of binary classification; and task b) will be referred to as *compound splitting*. The tasks can be seen as part of the more general problem of morphological segmentation, which refers to the splitting of a word into morphemes (affixes, roots, endings).

The following study constitutes, to the best of our knowledge, the first foray into automatic compound identification and compound splitting in Czech. Section 2 is a non-exhaustive overview of existing accounts of compounding relevant to this study. After a brief report on the compilation of the data set including examples of some challenges of Czech compounding (Section 3), the experiments are described and their performance is compared in Section 4. The solutions we implemented include a baseline solution which performs compound splitting only. A more advanced approach based on phonemic string similarity we call *Interlexical Matrices of Likeness*, or $IML()$, is also limited to compound splitting. Finally, a deep learning based tool dubbed *Czech Compound Splitter* was trained, which simultaneously carries out both compound identification and compound splitting. Section 5 contains the summary of this study.

2 Related work

2.1 Compounding in Czech

Theoretical descriptions of compounding in Czech are optimized for human readers. Bozděchová (1997) distinguishes two types of compounding in Czech, depending on whether the words entering the composition are formally modified or not. *Compounding proper*, which requires morphological adjustment of

the input words, and *compounding improper*, which is the result of simple concatenation of a syntactic phrase with no morphological adjustments. In addition, Bozděchová puts forth a multi-level classification, starting from the part-of-speech category of the output compound and then proceeding to semantic criteria (considering the meanings of the input items, of the output compounds and the relationship between the output and the inputs). Moreover, it is taken into account whether the compound is a result of composition only or whether also other word-formation processes (derivation, conversion) were at play. For instance, the compound adjective in (1) was coined through composition proper, when the ending *-ý* in the first input adjective (*tmavý* 'dark') was dropped and an *-o-* interfix was used to concatenate it with the second adjective (*modrý* 'blue'). In (2), the input adjective (*tvrdý* 'hard') undergoes a similar formal modification, but the second item (the noun *hlava* 'head') is converted into an adjective through replacing the nominal ending by an adjectival one (*hlava* 'head' → *-hlavý* 'headed', which cannot be used separately in Czech). Analogically to this example of compounding and conversion in one step, in (3) the compound is formed through compounding and derivation (i.e., the addition of the agent suffix *-ec* to the input verb). A straightforward example of composition improper is the concatenation of two nouns to a compound adverb in (4). A reversal of the ordering of the input words is permissible, resulting in the compound verb in (5).

(1) ***tmavý*** + ***modrý*** → ***tmav|-o-|modrý***
dark.ADJ blue.ADJ dark-blue.ADJ

(2) ***tvrdý*** + ***hlava*** → ***tvrd|-o-|hlavý***
hard.ADJ head.NOUN stubborn.ADJ

(3) ***černý*** + ***odít*** → ***čern|-o-|oděnec***
black.ADJ dress.VERB black dressed man.NOUN

(4) ***chvála*** ***Bohu*** → ***chvála|bohu***
praise.NOUN God.NOUN-DAT.SG thankfully.ADV

(5) ***přát*** ***blaho*** → ***blaho|přát***
wish.VERB wellness.NOUN-ACC.SG congratulate.VERB

In a recent paper on compounding in West Slavic languages, Ološtiak and Vojteková (2021) restrict themselves to non-native compounds, especially to compounds of partially or fully Greek-Latin origin (from here: *neoclassical compounds*). Four types of word-formation formants are distinguished, namely bases, baseoids, affixoids, and affixes. Bases are items that can appear freely and carry lexical meaning (*terapie* 'therapy', like in *ergoterapie* 'occupational therapy'); baseoids are items that do not appear freely, but carry lexical meaning regardless (*ergo-*, in *ergoterapie* 'occupational therapy'), and affixoids are items that are diachronically lexical, but have gradually lost their ability to appear independently and have generalized their meaning enough to effectively behave like derivational items. Three types of compounds are delimited according to the type of formants they involve. *Proper compounds*[1] are characterized as being composed of two bases (e.g. *sérum* 'serum' + *pozitivní* 'positive' → *séropozitivní* 'seropositive'). *Semi-compounds* are composed of one base and one baseoid (e.g. *krypto-* 'crypto-' + *politika* 'politics' → *kryptopolitika* 'cryptopolitics'). Finally, *quasi-compounds* are composed of two baseoids (e.g. *eko-* 'eco-' + *-logie* '-logy' → *ekologie* 'ecology').

Our conceptualization of neoclassical compounds is largely congruent with this classification, with a reduction in granularity. Everything they consider to be a *baseoid* and most of what the authors consider to be an *affixoid* is considered to be a neoclassical constituent by us. This creates a small amount of inconsistency in exchange for increased simplicity and reduced granularity. For instance, we consider the formant *-pidi-* (considered an affixoid by the authors) to be a neoclassical constituent, because it behaves almost exactly the same way as *-mini-*, with regards to both semantics and behaviour within word formation. We prefer this interpretation despite the fact that it is traced back to the Czech noun *píď* 'span' (unit of length). We also systematically interpret neoclassical constituents as identical whenever their etymology and semantics allow for it, even under circumstances where they undergo formal changes. For instance, the first element of *logografie* 'logography' (*logo-*) and the second element of *sociologie*

[1]The usage of this term by these authors is distinct from Bozděchová's proposal above.

'sociology' (*-logie*) are seen to be the same, since they both ultimately descend from the same Greek root. In our data, they are represented by the string *-log-*, cf. Section 3.2 for more details.

Štichauer (2013) presents an attempt to classify Czech compounds using three levels of categorisation, akin to the way Romance compounds are handled by Bisetto and Scalise (2005). The first level is the distinction between *coordinative*, *subordinative* and *attributive* compounds. The second level distinguishes between *exocentricity* and *endocentricity*, or *headedness* – in other words, whether or not the compound has a semantic head. The third level distinguishes between every possible combination of part-of-speech category of the input words and the part-of-speech category of the output compound in the format $[X + Y]_Z$, where X and Y stand for the input part of speech and Z stands for the part of speech of the resulting compound.

2.2 NLP approaches toward compounding

Czech has neither a static word formation data resource with a notable amount of parent-linked compounds nor a procedural tool for identifying or splitting compounds. Derivational Analyzer of Czech (Derivancze; Pala and Šmerk, 2015), as its name suggests, is limited to derivational relations in the lexicon of Czech. Another word-formation resource for the language, DeriNet, maps derivation by means of linking words to the words they are respectively derived from all the way to their roots. DeriNet, in spite of its name, is additionally equipped for handling compounding as well, in that its data format allows for a single lexeme to have multiple parents. DeriNet version 2.0 (Vidra et al., 2019) contained $33,932$ lexemes identified as compounds, out of which $1,252$ had their respective parent words identified. The work on this paper has contributed to the release of DeriNet version 2.1 by identifying the parents of $1,439$ compounds. The new version therefore contains a total of 2691 compounds with identified parents. (Vidra et al., 2021)

The situation regarding the computational handling of compounds is different in some other languages. Specifically, GermaNet (Hamp and Feldweg, 1997) contains nearly $100,000$ split nominal compounds, and CELEX (Baayen et al., 1996) includes $71,249$ split compounds for Dutch, $12,853$ split compounds for English and $19,768$ for German, all done manually.

Procedural compound splitting has been successfully demonstrated to be feasible in several languages. Henrich and Hinrichs (2011) linked German nominal compounds to their respective parents in GermaNet using an ensemble of pattern-matching models with an accuracy of 92%. Sugisaki and Tuggener (2018) achieved an F1-score of 92% for finding split-points in German compounds using an unsupervised approach, although they also restricted their efforts to noun-headed compounds only. Ma et al. (2016) achieved an accuracy of 95% using a neural approach trained on the aforementioned GermaNet. Their model performed both splitting and identification of compounds, with the accuracy being an aggregated score of both. Krotova et al. (2020) achieved an accuracy of 96% with a deep neural model trained on GermaNet data, again restricting themselves to nominal compounds.

A significant amount of research has been dedicated to the study of Sanskrit compounds. This ranges from early, relatively simple rule-and-lexicon based attempts by Huet (2005), who lists no accuracy in his study, to Hellwich and Nehrdich's (2018) deep-learning solution trained on a corpus of 560,000 Sanskrit sentences with its compound split points annotated, achieving an accuracy of 96%.

As for other languages, Clouet and Daille (2014) achieved F1-scores for finding split-points in English and Russian compounds of 80% and 63% respectively, using a corpus-based statistical approach on manually split compounds. Russian is important for this study, because it is a Slavic language like Czech and thus this result is the most comparable to the ones presented here.

3 Compilation of the data set

3.1 Challenges

What follows is a qualitative analysis of some formal difficulties that regularly appear in Czech compounding. Please note how the phenomena often accumulate within the same word, and that the list is not by any means exhaustive. None of these nor any similar difficult cases were dropped from the data.

For data-based approaches, the simplest case seem to be compounds formed by simple concatenation (cf. compounding improper in the literature discussed above). For instance, the adjective in (6) corresponds

directly to the syntactic phrase *vždy zelený* ‘always green’. In (7), neither input word undergoes any morphological change during the composition, which is characteristic for composition improper, but the output noun cannot be associated with no such phrase, which is typical of composition proper. From the perspective of algorithmic splitting, however, the two compounds are very much alike, in that the procedure of finding their parents consists merely of finding the appropriate split point.

(6) ***vždy*** ***zelený*** → ***vždy|zelený***
always.ADV green.ADJ-NOM.SG evergreen.ADJ

(7) ***garáž*** + ***mistr*** → ***garáž|mistr***
garage.NOUN master.NOUN garage supervisor.NOUN

An interfix is added between the two input words in other compounds, usually *-o-* or *-i-*. This interfix replaces the inflectional ending of any non-final parent; cf. the ending *-a* in the feminine noun *ryba* ‘fish’ in (8). Additionally, stem allomorphy often appears. It may takes the form of vowel alternation, for example /e/ → ∅, like in (9).

(8) ***ryba*** + ***lov*** → ***ryb|-o-|lov***
fish.NOUN hunt.NOUN fishery.NOUN

(9) ***krev*** + ***tok*** → ***krv|-o-|tok***
krev.NOUN flow.NOUN bloodflow.NOUN

As mentioned above, compounding and conversion (or derivation) in one step is possible, as exemplified above and here in (10). Stem alternation may take place, like in (11), where a case of stem vowel alternation (/e/ -> ∅ and /e:/ → /o/), a stem consonant alternation (/s/ → /d/), an interfix, compounding and conversion in one step all occur at the same time. Note that an alternative analysis of the compounds in (8) and (9) can be proposed that would parallel (11): *krev* ‘blood’ + *téct* ‘to flow’ → *krvotok* ‘bloodflow’, *ryba* ‘fish’ + *lovit* ‘to hunt’ → *rybolov* ‘fishery’. In the data we use in our experiments, both analyses are captured (see Section 3.2).

(10) ***modrý*** + ***oko*** → ***modr|-o-|oký***, but no ****oký***
blue.ADJ eye.NOUN blue-eyed.ADJ

(11) ***pes*** + ***vést*** → ***ps|-o-|vod***, but no ****vod***
dog.NOUN lead.VERB dog handler.NOUN

In (12), the compound is traced back to the noun phrase *chtivý holek* ‘wanting of girls’, with its original ordering switched. Additionally, there are compounds that cannot be meaningfully split into two parents; cf. the compound in (13) which is composed of a multi-word numeral expression (*dvě a půl* ‘two and a half’) and the final part which was converted from a noun (*léto* ‘year.NOUN’ → *-letý* ‘-year.ADJ’).

(12) ***chtivý*** ***holek*** → ***holek|chtivý***
wanting.ADJ girl.NOUN.GEN.PL wanting girls.ADJ

(13) ***dvě*** + ***a*** + ***půl*** + ***léto*** → ***dva|a|půl|letý***, but no ****letý***
two.NUM and.CONJ half.NUM summer.ADJ two-and-a-half-year-old.ADJ

The so-called *neoclassical compounds* constitute what Ološtiak and Vojteková (2021) consider semi-composition and quasi-composition. The noun *sociologie* ‘sociology’ in (14) is an example of quasi-composition in this framework. In a broader sense, chemical compounds satisfy the definition of semi-composition, as in (15).

(14) ***-soci-*** + ***-log-*** → ***soci|-o-|logie***, but no ****-soci-*** nor ****-log-***
-soci-.NEOCON -log-.NEOCON sociology.NOUN

(15) ***-tetra-*** + ***chlor*** + ***ethylen*** → ***tetra|chlor|ethylen***, but no ****tetra***
-tetra-.NEOCON chlorine.NOUN ethylene.NOUN tetrachlorethylene.NOUN

3.2 Manual annotation of DeriNet data

The compilation procedure began by extracting $1,500$ words from the DeriNet word-formation resource that had previously been labelled as having compound status. As their parent words had not been yet identified, so this had to be done by hand. 53 were dropped, because some had been labelled as compounds mistakenly (*levopimar*, a medicine brand name), or are derivatives of compounds (e.g. the adverb *velechytře* derived from the adjective *velechytrý* 'very clever'). After this cleanup process was done, $1,447$ compounds remained in the data set. 20% of the data set compounds was held out for the purposes of validation. The training set therefore consisted of $1,158$ hand-annotated compounds, while the *holdout data set* set consisted of 289 hand-annotated compounds. The holdout set was further split in half. The first half, the *test set*, was used to determine when to stop training *Czech Compound Splitter*. The performance of all the approaches presented here was evaluated on the other half, the *validation set*.

Neoclassical constituents, as they do not have an agreed-upon citation form, are labelled with hyphens on both sides, maintaining the original Greek stem as bare as possible. In order to reflect the consistency described in Section 2.1, we label both the second constituent of *sociologie* 'sociology' and the first constituent of *logografie* 'logography' as *-log-*. We keep the 'o', if it resolves an otherwise arising ambiguity. For example, we label the first element of *bigamie* 'bigamy' as *-bi-* and the first element of *biologie* 'biology' as *-bio-*, preferring this slight annotation inconsistency over label ambiguity. Greek orthography is respected as much as possible, so we respect the distinction between τ and θ, so the first element of *teologie* 'theology' is labeled as *-theo-* (not *-the-*, as that would be ambiguous with the root of *teorie* 'theory'). Zero ablaut forms are preferred as labels of neoclassical compounds, unless this would result in an asyllabic label. Thus, both the first element of *gastronomie* 'gastronomy' and the second element of *melanogaster* (the epithet of the fruitfly *Drosophila melanogaster*) is labeled as *-gastr-*, but the first element of *gonokok* 'gonococcus' and the second element of *mutagen* 'mutagen' are labelled as *-gen-*.

The first two of the three algorithmic solutions require a lexicon to find potential parent-candidates in. DeriNet was used as a basis for this lexicon, but it restricts itself to nouns, verbs, adjectives, and adverbs. Lexemes of other part-of-speech categories were extracted from a Czech inflectional dictionary (MorfFlex; Hajič et al. 2020) and added into the lexicon. Finally, all neoclassical constituents identified during the manual annotation were added into the lexicon.

3.3 Generation of synthetic data

Because the hand-annotated data set of compounds obtained from DeriNet is too small to reliably train a deep learning model, we simulated various compound formation procedure that take place in Czech. For example, in (16) we see the process of taking a random adjective stripped of its ending and concatenating it with an *-o-* interfix and with another random adjective. The output is usually nonsensical, like in the example, but formally correctly formed.

(16) **Adjective 1** + ***-o-*** + **Adjective 2** → **Compound Adjective**
důležitý + ***-o-*** + ***neomylný*** → ***důležitoneomylný***
important.ADJ infallible.ADJ important-infallible.ADJ

For the purposes of training *Czech Compound Splitter*, we simulated a number of such compound formation procedures in Python using randomly selected lexemes from DeriNet, creating a data set of about $280,000$ synthetic compounds. The compound part of the training data set therefore consisted of this synthetic data set combined with all of the hand-annotated compounds apart from the holdout described in Section 3.4.

3.4 Evaluation methodology

For about 38% of the hand-annotated compounds in our dataset, there was ambiguity as to which parents they should be linked to. For instance, *monoprogramový* 'having a single programme' may be considered to be either composed of the neoclassical constituent *-mon-* and the adjective *programový*,

or it alternatively may be composed of *-mon-* and the noun *program*, which would be derivation and compounding in one step. For the purposes of evaluation, both were considered to be correct splittings.

Additionally, a more relaxed metric was proposed which considers a predicted parent-candidate to be correct if it belongs to the same morphological family as the annotated parent. This metric is referred to as *root accuracy*, because all items of a morphological family are represented as a tree structure with the unmotivated word as the root node in DeriNet. DeriNet data are used to determine whether or not the predicted parent-candidate shares the same morphological family as the annotated parent. The solutions described in the following section exhibit different weaknesses and strengths.

4 Experiments

4.1 Baseline solution

This is a naive algorithm only intended as a baseline to help provide context for the performances of the other solutions. This solution assumes the given compound has two parents. It attempts to find an 'o' grapheme in the middle third of the input word. If it finds one, it splits the word on this 'o', creating two subwords. If no 'o' is found, it does the same with 'i'. If no 'i' is found, it simply splits the input in the middle, if the number of graphemes in the graphical word is even, the left subword ends up being the longer one. Between each subword and every word in the lexicon, Levenshtein (1966) distance is calculated, and the word with the smallest distance from the subword is selected. Please refer to Table 4 to see its performance.

4.2 The $IML()$-based heuristic algorithm

The second attempt to split compounds is based on a phonological similarity measurement function developed specifically for this purpose. We developed a function that takes two words as input and returns a rational number representing the total degree of phonological similarity between the two words. We then attempted to find pairs of words which, when concatenated, exhibited a low degree of $IML()$ similarity with the compound in question. $IML()$ cannot perform compound identification, because the method already assumes the input word has exactly two parents.

4.2.1 The $IML()$ matrix function

We began by manually defining a phonemic correspondence weight by hand for each possible pair of phonemes in Czech. The minimum weight is 0, which is the correspondence weight strictly between a phoneme and itself, and the maximum weight is 1, which is the correspondence weight between a phoneme and a phoneme it never alternates with, like between /a/ and /t/. Note that this relationship is asymmetric by design, because we estimated that, for example, /h/ $\rightarrow$ /z/ is much more common than /z/ $\rightarrow$ /h/. From this, it directly follows that the ordering of the words that are input into the $IML()$ function matters. There are 32 phonemes in the Czech language, so it follows that the total amount of phonemic correspondences equals $32^2 = 1024$. This can be described by a square matrix, where each column and row corresponds to one of the Czech phonemes and each element describes the correspondence weight between the Czech phonemes. This is what we call a *correspondence matrix*. Part of the matrix used in this study is shown in Table 1. Note that the diagonal is composed entirely of zeroes, and that the matrix is not symmetric with respect to said diagonal, which reflects the asymmetric nature of Czech phoneme alternation described in the previous paragraph.

The $IML()$ similarity measurement function takes two words, transcribes both of them phonologically, and uses the values found in the *correspondence matrix* to build a separate matrix of correspondence weights between every single pair of phonemes from the two input graphical words. The cheapest path through it is found, beginning in the top left corner of the matrix, and ending in the bottom right corner. We used the $A*$ algorithm, an extension of Dijsktra's algorithm, to find the shortest path (Hart et al., 1968).

The lower the output value, the higher the similarity, with $IML(word_1, word_2)$ being equal to 0 if and only if $word_1 = word_2$, because the correspondence weight between a pair of phonemes is zero if and

	/t/	/n/	/r/	/s/	/z/	/ts/	...
/t/	0	0.8	1	0.8	0.9	0.8	...
/n/	0.7	0	0.9	1	1	1	...
/r/	0.7	0.9	0	0.9	1	1	...
/s/	0.6	1	0.7	0	0.2	0.6	...
/z/	0.8	0.3	0.9	0.2	0	1	...
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱

Table 1: *S*ample of the referential matrix of correspondence weights between pairs of Czech phonemes.

IML(*černomodrý*,*černý* + *černý*) = 5.8
IML(*černomodrý*,*černý* + *červený*) = 5.0
IML(***černomodrý***,***černý* + *modrý***)= **0.6**
IML(*černomodrý*,*červený* + *černý*) = 5.7
IML(*černomodrý*,*červený* + *zelený*) = 6.9
IML(*černomodrý*,*červený* + *modrý*) = 2.6
IML(*černomodrý*,*modrý* + *černý*) = 9.6
IML(*černomodrý*,*modrý* + *zelený*) = 9.8
IML(*černomodrý*,*modrý* + *modrý*) = 10.6

Table 2: Sample of the algorithm's functioning, without the heuristic filter.

only if the two phonemes in the pair are identical – which is why the diagonal of the *referential matrix* is composed of zeros, and in that the only zeros in the *referential matrix* are located on the diagonal.

4.2.2 The heuristic

Based on this similarity function, we were able to find the pair of words from the lexicon mentioned previously which, when concatenated, exhibited the highest similarity with the compound word in question. The algorithm therefore requires the compound in question and a lexicon to find its parents in. A visual demonstration of the idea behind the algorithm with the word *černomodrý* 'black and blue' and a toy lexicon can be viewed in Table 2. The table shows the outputs of the $IML(compound, word_1 + word_2)$ calculations for each word pair from the {*černý*, *červený*, *modrý*} {'black', 'red', 'blue'} lexicon. The algorithm generates all pairs of lexemes from a given lexicon, concatenates them and calculates $IML()$ for each pair. It then (correctly in this case) selects the pair with the smallest value. The problem is that the size of our lexicon ultimately exceeded $800,000$ lexemes, meaning that every time a compound is split, over $800,000^2 = 6.4 \times 10^{11}$ interlexical matrices need to be built and run through the $A*$ pathfinding algorithm.

A heuristic filter was therefore added. For this purpose, a variant of the $IML()$ function, the $IML_{sub}()$ function, was defined. The two functions are similar with two key differences. First, in the case of the $IML_{sub}()$, the cheapest path does not have to reach the bottom right corner of the matrix. Instead, the path's total cost is calculated whenever it reaches either the right or bottom edge of the *interlexical matrix*. $IML_{sub}(word_1, word_2)$ returns the *degree* to which $word_2$ is a fuzzy substring of $word_1$, with respect to their phonological similarity. Second, the pathfinding algorithm used in $IML_{sub}()$ is not $A*$, but a best-first solution. This makes $IML_{sub}()$ significantly faster than $IML()$, because the whole *interlexical matrix* need not be constructed beforehand. Only word pairs $(lexeme_1, lexeme_2)$ which satisfied the following conditions were selected:

1. $First2Chars(lexeme_1) = First2Chars(compound)$,
2. $CountSyl(lexeme_1 + lexeme_2) \geq CountSyl(compound)$,
3. $IML_{sub}(lexeme_1) \leq 2.2 \ \ \& \ \ IML_{sub}(lexeme_2) \leq 2.2$,

where $First2Chars()$ is a function which returns the first two graphical characters of a given graphical word, $CountSyl()$ counts the syllables of the given graphical word (assuming it is a Czech word) and

Compound	English translation	*CCS* (incorrect) splitting	Correct splitting
dlouhohořící	'long-burning'	*dlouhohořící*	*dlouho* + *hořící*
CCS returns the original string, performing no splitting.			
osmiramenný	'eight-armed'	*osm* + *ramenný*	*osm* + *rameno*
CCS returns a non-existing derivative of an existing word.			
petrogeneze	'petrogenesis'	*-petro-* + *geneze*	*-petr-* + *geneze*
CCS includes the interfix in one of the parents.			

Table 3: A sample of the errors *Czech Compound Splitter* (CCS) typically makes.

$compound$ is the input compound being split.

4.2.3 Output evaluation

This pair of words then constituted the predicted parents. The performance of this method in compound splitting can be found in Table 4. The application of the algorithm seems to be much less practical than that of *Czech Compound Splitter*, because it takes about ten to fifteen minutes to split a single compound on a single processor given a lexicon of our size, despite the fact that the algorithm's asymptotic time complexity (even without the heuristic) is $O(n) = n^2$, where n refers to the size of the lexicon. The matrix building step takes $|word1| \times |word2|$ correspondence matrix lookup operations, but because the step occurs exactly once for each parent-candidate pair, it constitutes a constant, and is therefore by convention omitted when assessing asymptotic time complexity. It is additionally of interest that the root accuracy of this method was higher by 11 percentage points than its raw accuracy. Error analysis revealed that this increase is primarily caused a common error where a substring of a compound is homonymous to a noun derived from an adjective, while that adjective is the parent. For example, *bíločerný* 'black and white' is split into the noun *bílo* 'whiteness' and the adjective *černý* 'black', while two adjectives (*bílý* 'white' and *černý* 'black') are the correct parents.

4.3 Czech Compound Splitter

Because the performance and practicality of the $IML()$-based heuristic algorithm was deemed unsatisfactory, a neural compound splitting tool we named *Czech Compound Splitter* was created. It decides if an graphical word is a compound and if so, it returns its predicted parent words, all in one step. If the graphical word is identified as a compound, it returns its parents separated by spaces. The estimated number of parents is thus the number of spaces in the output $+1$, and the status of a compound is determined if this number is greater than 1.

The tool was created by using the Marian machine translation framework developed by Microsoft (Junczys-Dowmunt et al., 2018) to build a model and train it. This was done by feeding the model a parallel corpus of input and output data, where the model is trained to take an element of the input data, which was a Czech word, and the output was either the single derivational parent of that word if it was a non-compound, or all of the parents of that word separated by spaces if it was a compound. For example, *Czech Compound Splitter* was trained to return *kov* 'metal' upon being given the graphical word *kovový* 'made of metal', and to return *uhlík vodík* 'carbon hydrogen' upon being given the graphical word *uhlovodík* 'carbohydrate'. The non-compounds and their parents were taken from DeriNet.

The total training data set for *Czech Compound Splitter* consisted of:

- $1,164$ genuine compounds, with their splittings
- $280,000$ synthetic compounds, with their splittings
- the near entirety of DeriNet's non-compounds, with their derivational parents

The rest of DeriNet's non-compounds, totalling 144 lexemes, was held-out in order to test the performance of *Czech Compound Splitter* in compound identification.

Method	1st parent accuracy	2nd parent accuracy	Overall accuracy	Overall root accuracy
Baseline	22%	42%	11%	16%
$IML()$	42%	66%	24%	39%
Czech Compound Splitter	61%	66%	54%	61%

Table 4: Overall performances the three solutions exhibited.

4.3.1 Model evaluation

In compound identification, *Czech Compound Splitter* achieved an accuracy of 92% and an F1-score of 91%. Its performance in compound splitting can be found in Table 4. We see that root accuracy is just barely higher than accuracy. Error analysis reveals that this was due to the fact that a large proportion of the mistakes *Czech Compound Splitter* made because it often did not recognize the input as a compound. Similarly, it frequently returned a nonsensical string that is not a Czech word; see a sample of errors in Table 3.

It is worth noting that *Czech Compound Splitter* made only a single false positive error, meaning that it almost never labelled a non-compound as a compound. This suggests that it primarily recognizes compound status by detecting lexical-seeming substructures, as opposed to focusing on surface-level criteria like character length or the presence of an *-o-* interfix. *Czech Compound Splitter* run on a single GPU takes about 0.2 seconds to perform a single identification and splitting. The entire compiled model is about 300 MB in size, making its distribution as a Python package feasible, especially since it can be compiled to run on CPUs as well.

5 Conclusions

We present the results of the first attempts to automatically identify and split Czech compounds. While there has been a lot of attention invested into automatic compound splitting in languages such as German or Sanskrit, in the Slavic languages, the topic has largely, though not completely, been overlooked. We have attempted to tackle the problem using three approaches – one that uses simple heuristics, another based on an asymmetric word similarity metric based on finding the shortest path through a matrix of elements representing phonological similarity, and another utilizing a deep learning model partially trained on synthetic data. Despite a high degree of irregularity in Czech compounding, the *Czech Compound Splitter* tool achieved an accuracy of 54% in the task of compound splitting and an accuracy of 92% in compound identification. The work on this study has contributed to the creation of DeriNet version 2.1 by manually identifying the parents of 1,439 compounds, totalling 2,691 compounds with identified parents.

Acknowledgments

This work was supported by the Grant No. GA19-14534S of the Czech Science Foundation, the LINDAT/CLARIAH-CZ project of the Ministry of Education, Youth and Sports of the Czech Republic (project LM2018101), and by the Grant No. START/HUM/010 of Grant schemes at Charles University (Reg. No. CZ.02.2.69/0.0/0.0/19_073/0016935).

References

Harald R. Baayen, Richard Piepenbrock, and Leon Gulikers. 1996. The CELEX lexical database (CD-ROM). *Linguistic Data Consortium* .

Laurie Bauer. 2003. *Introducing linguistic morphology*. Edinburgh University Press.

Antonietta Bisetto and Sergio Scalise. 2005. The classification of compounds. *Lingue e linguaggio* 4(2):319–332.

Ivana Bozděchová. 1997. *Tvoření slov skládáním*. Institut sociálních vztahů.

Elizaveta L. Clouet and Béatrice Daille. 2014. Splitting of compound terms in non-prototypical compounding languages. In *Workshop on Computational Approaches to Compound Analysis*. pages 11–19.

Jan Hajič, Jaroslava Hlaváčová, Marie Mikulová, Milan Straka, and Barbora Štěpánková. 2020. MorfFlex CZ 2.0. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University. http://hdl.handle.net/11234/1-3186.

Birgit Hamp and Helmut Feldweg. 1997. GermaNet – a lexical-semantic net for German. In *Automatic information extraction and building of lexical semantic resources for NLP applications*. pages 9–15.

Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. 1968. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics* 4(2):100–107. https://doi.org/10.1109/TSSC.1968.300136.

Oliver Hellwig and Sebastian Nehrdich. 2018. Sanskrit word segmentation using character-level recurrent and convolutional neural networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. pages 2754–2763.

Verena Henrich and Erhard Hinrichs. 2011. Determining immediate constituents of compounds in GermaNet. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing 2011*. pages 420–426.

Gérard Huet. 2005. A functional toolkit for morphological and phonological processing, application to a Sanskrit tagger. *Journal of Functional Programming* 15(4):573–614.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. 2018. Marian: Fast neural machine translation in C++. In *Proceedings of ACL 2018, System Demonstrations*. pages 116–121. https://arxiv.org/abs/1804.00344.

Irina Krotova, Sergey Aksenov, and Ekaterina Artemova. 2020. A joint approach to compound splitting and idiomatic compound detection. In *Proceedings of the 12th Language Resources and Evaluation Conference*. pages 4410–4417. https://aclanthology.org/2020.lrec-1.543.

Vladimir I. Levenshtein. 1966. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics-Doklady* 10(8):707–710.

Jianqiang Ma, Verena Henrich, and Erhard Hinrichs. 2016. Letter sequence labeling for compound splitting. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*. pages 76–81.

Martin Ološtiak and Marta Vojteková. 2021. Kompozitnosť a kompozícia: príspevok k charakteristike zložených slov na materiáli západoslovanskỳch jazykov. *Slovo a slovesnost* 82(2):95–117.

Karel Pala and Pavel Šmerk. 2015. Derivancze—derivational analyzer of Czech. In *International Conference on Text, Speech, and Dialogue*. pages 515–523.

Pavel Štichauer. 2013. Je možná nová klasifikace českỳch kompozit? *Časopis pro moderní filologii* 95(2):113–128.

Kyoko Sugisaki and Don Tuggener. 2018. German compound splitting using the compound productivity of morphemes. In *14th Conference on Natural Language Processing*. pages 141–147.

Jonáš Vidra, Zdeněk Žabokrtský, Lukáš Kyjánek, Magda Ševčíková, and Šárka Dohnalová. 2019. DeriNet 2.0. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University. http://hdl.handle.net/11234/1-2995.

Jonáš Vidra, Zdeněk Žabokrtský, Lukáš Kyjánek, Magda Ševčíková, Šárka Dohnalová, Emil Svoboda, and Jan Bodnár. 2021. DeriNet 2.1. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University. http://hdl.handle.net/11234/1-3765.