Show, Don't Tell: Visualising Finnish Word Formation in a Browser-Based Reading Assistant

Frankie Robertson
University of Jyväskylä
frankie@robertson.name

Abstract

This paper presents the *NiinMikäOli?!* reading assistant for Finnish. The focus is upon the simplified presentation and visualisation of a wide range of word-level linguistic phenomena of the Finnish language in a unified form so as to benefit language learners. The system is available as a browser extension, intended to be used in-context, with authentic texts, in order to encourage free reading in language learners.

1 Introduction

This paper presents an intelligent reading assistant for Finnish. The system, *NiinMikäOli?!* (English: *TheWhatNow?!*), presents word and idiom definitions in-context. The system can be used through a web interface either as a dictionary or by manually entering or copying text into a text field, or ideally, as a browser extension to assist with reading Finnish web pages. When used as a browser extension, *NiinMikäOli?!* presents word definitions in a sidebar.

There is increasing interest in contextualised learning of vocabulary (Godwin-Jones, 2018), and *NiinMikäOli?!* aims to facilitate this in the context of web pages. *NiinMikäOli?!* can be classified as an ATICALL (Authentic Text Intelligent Computer Aided Language Learning) system, defined by Meurers et al. (2010a) as software which produces enhanced input based on real texts.

The focus of this paper is upon *NiinMikäOli?!*'s simplified, unified view of the Finnish language, which uses information visualisation techniques to "show rather than tell" learners about morphological and word formation features. A principle aim is to avoid presentations which tell learners about word-level grammatical features such as technical linguistic language including Latinate names for nominal cases, rather opting to highlight their surface forms. The user interface visualises the connection between surface forms, analytic forms and definitions.

Described first is the construction of the baseline reading assistant system. The system is by and large similar to existing systems such as GLOSSER (Nerbonne et al., 1998) or the reading assistant features of SMILLE (Zilio et al., 2017) or Revita (Katinskaia et al., 2017) – or even very widely used systems such as the WordNet-based alternative translations shown when a single word is selected in Google Translate. Notable as an improvement over some of those systems is *NiinMikäOli?!*'s use of a full scale Word Sense Disambiguation (WSD) system. The rest of this paper describes the motivation behind and implementation of *NiinMikäOli?!*'s visualisations and its exhaustive treatment of complex lexical items such as derived words, compounds and multiword expressions.

2 Baseline system

A combined lexical resource of Finnish was created by combining two sources: FinnWordNet (Lindén and Carlson, 2010) and Wiktionary. The Wiktionary definitions were extracted from publicly available dumps, using a Python script[1] . The Python script parses MediaWiki markup into word senses using `mwparserfromhell`[2].

At least one definition was extracted from 99.8% of a total of 153 196 Wiktionary pages con-

[1] Available at https://github.com/frankier/wikiparse.

[2] https://github.com/earwig/mwparserfromhell

Frankie Robertson 2020. Show, don't tell: Visualising Finnish word formation in a browser-based reading assistant. *Proceedings of the 9th Workshop on Natural Language Processing for Computer Assisted Language Learning (NLP4CALL 2020)*. Linköping Electronic Conference Proceedings 175: 37–45.

taining *Finnish* as a section heading[3]. Of these, 90 653 are lemmas rather than inflected forms. For comparison, FinnWordNet contains 139 871 headwords, which are mostly lemmas but include occasional idiomatic word forms such as humalassa (literally "in hops").

FinnWordNet is modelled after WordNet, and as such has very fine-grained sense distinctions. This results in potentially overwhelming the learner with too much information. Furthermore, some Wiktionary senses are likely to essentially duplicate FinnWordNet. Thus, similar definitions should be clustered together and only the best definition displayed by default.

The clustering and alignment was created using affinity propagation (Frey and Dueck, 2007). The distances graph is constructed by taking cosine distances between definitions, represented as vectors based on the English text of their definitions according to the English sentence similarity model of Reimers and Gurevych (2019). This model is based on pretrained English BERT models fine-tuned on a semantic similarity task. The pretrained `bert-large-nli-stsb-mean-tokens` model is used. Links between Wiktionary definitions with distinct etymologies are then removed and extra weight is given to Wiktionary definitions so as to encourage them to become exemplars of clusters. The resulting system obtains adjusted rand index scores of 0.48 on a gold standard of WordNet verbs grouped by PropBank sense obtained from Predicate Matrix (de Lacalle et al., 2016), and a score of 0.52 on a manually created clustering of 128 Wiktionary and WordNet noun definitions. The scikit-learn (Pedregosa et al., 2011) implementation of affinity propagation is used.

WSD is performed using UKB (Agirre et al., 2014). Since UKB is a graph based WSD algorithm, it only operates on definitions from FinnWordNet, which are connected by the semantic links from Princeton WordNet. In order to compare WordNet definitions with Wiktionary definitions, the clustering is used. Clusters are then ranked using their best WordNet definition as a representative. Since Wiktionary definitions are usually better for learners, they are pushed to the top of each cluster in the user interface.

[3]In a Wiktionary dump from 6/4/2019.

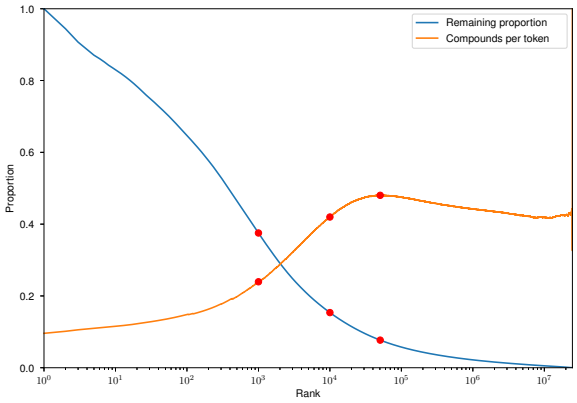

Figure 1: Proportions related to words unknown to a simplified model of a language learner. The x-axis gives the rank of the word that the learner has learned all words up to. Remaining proportion is the proportion of words in running text unknown to the language learner. Compounds per token is the proportion of unknown words which are compounds.

3 Linguistic rationale

Finnish is morphologically rich[4]. Substantives are declined for case and number and verbs are conjugated for person, tense and voice. Finnish word formation is also rich[5]. It includes a number of highly productive derivational morphemes, including many deverbal morphemes which is characteristic of the language. Compounding also plays a major role in Finnish word formation, with many of the compounds being semantically transparent. Finnish also has a number of enclitic particles such as the question forming "-ko/-kö". Finally, it has MWEs (Multi-Word Expressions) such as idioms. In Finnish these may take the form of syntactic frames, treated here as gapped MWEs e.g. "pitää ___-sta", which could occur in a form such as "pidän voileipäkakusta" (English: I like sandwich cake) distinguished from e.g. "pidän voileipäkakun" (English: I keep sandwich cake).

Why bother going to the effort of making a comprehensive treatment of word formation and complex word types? After all, these lexical items occur relatively infrequently in running text and so it may seem like a poor allocation of effort to spend time dealing with them. One assumption here is that these elements become more important after the beginner stage of language learning. If we assume a very simplified model of lexical acquisition where words are learnt in de-

[4]See for example Karlsson (2015).
[5]See for example Hyvärinen (2019).

scending order of frequency, we can analyse properties of words that the language learner does not know and therefore may like to look up. Figure 1 shows two such properties varying as the number of words the learner knows increases: the proportion of all words seen which are unknown, and the proportion of unknown words which are compounds. The data is based on 1.5 billion tokens of analysed Finnish text from the Turku Internet Parsebank (Laippala and Ginter, 2014). Taken as a whole, the corpus is 9.8% compounds. Supposing that an intermediate learner may know somewhere between 1000 and 10 000 words. After learning 1000 words, 24% of unknown words would be compounds, and after 10 000, it would be 42%. Thus quite a large proportion of words unknown to intermediate level learners are compounds. It is assumed that other complex lexical items such as MWEs follow a similar pattern. Here we refer to any item which can be given a definition, including lemmas, individual morphemes and MWEs as *headwords*.

Admitting these items are frequent, the next question becomes, why is simple lemma extraction not sufficient? One argument against performing lemma extraction and simply showing lemmas comes from the noticing hypothesis (Schmidt, 1990) which states that without attention to form (Lightbown and Spada, 2013, pp. 168–175), second language learners in particular are prone to not acquiring fine-grained grammatical knowledge. Following this concept, systems such as those of Meurers et al. (2010b) and Reynolds et al. (2014) were created to automatically enhance input in web pages in order to promote noticing of, for example, parts of speech. *NiinMikäOli?!* follows a similar direction, but instead focusses on morphemes, drawing attention to the connection and overlap between analytic and surface forms, so to promote learning of morphology, as well as attention to the formation of the word itself.

Why analyse words using only normalised segments, rather than — as a lot of reference material for the Finnish language does — using grammatical descriptions, such as Latinate names for case endings. The reason for this is twofold. Firstly, as Bleyhl (2009) notes, treatments of language which are heavy on grammatical analysis and the associated linguistic terminology can be counter-productive in language instruction since they draw attention away from the comprehensible input needed for true language acquisition. This large amount of extra material can lead to reduced confidence from learners. Secondly, due to Finnish's agglutinative morphology, contrasted with a fusional language like Latin, it is simply not necessary to add this extra layer of analytical language, since many Finnish inflectional morphemes occur in the same form or an easily recognisable form at all times, and they can thus be referred to by their normalised form. Consider for example, the Finnish system of locative case endings. These have a fairly good correspondence in terms of function with prepositions in English. Imagine if, when teaching English, every preposition was also given a name to describe it so that we would always refer to "from" as "the elative preposition". It is hard to imagine that a learner would be well served by this extra indirection! This principle is somewhat flexible, however, and the names of the most common case endings — partitive and genitive — are shown on the basis that their usage is more grammatical. They are more often obligated by context rather than used with the intention of conveying extra information. Plural is referred to by-name since it is likely to be familiar.

4 Implementation

The pipeline from running text to analytical segments, described in this section, is shown in Figure 2.

4.1 MWE lexicon & extraction

FinnMWE (Robertson, 2020) is used as a lexicon of MWEs. In order to extract MWEs from running text, for each MWE is indexed by all possible lemmas of a key token. In case the head is known, it is used as the key token, otherwise the rarest token based on wordfreq (Speer et al., 2018) is used. MWEs are then extracted from dependency trees obtained using the Turku neural dependency parsing pipeline (Kanerva et al., 2018). First, all key matches are found simultaneously by looking up all lemmas in the dependency tree. These candidates are filtered by trying to match each remaining token in the MWE against any neighbour, extending the neighbourhood in the process until the whole MWE is matched or it is impossible to proceed. When the MWE key is its head, its parent is excluded from the neighbourhood of potential

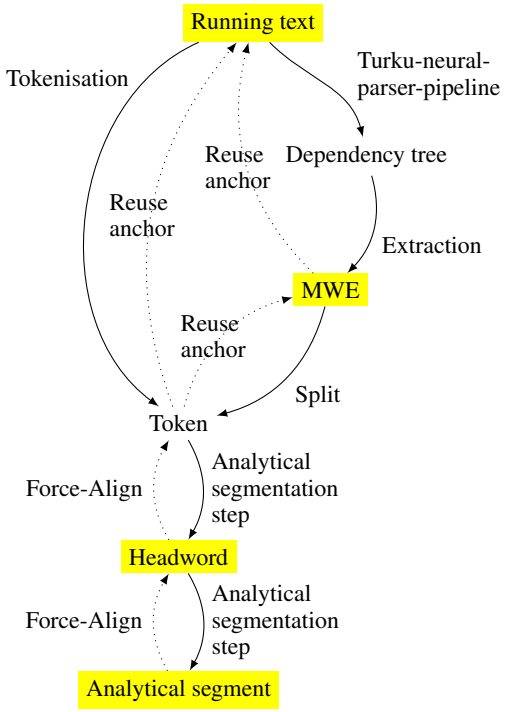

Figure 2: Diagram showing processing pipeline from surface forms to analytical segments. Objects indicated in yellow are linked within the user interface during the hover brushing interaction. Dotted lines indicate how spans corresponding to the in-node are found in the out-node.

matching tokens. MWE tokens without a lemma act as wildcards, and can match multiple tokens, but they must be connected within the dependency tree.[6]

4.2 Analytical segmentation data

The approach to analytical segmentation pursued here is to combine analyses from the Omorfi morphological analyser (Pirinen, 2015) and information from Wiktionary together to produce analytical segmentations. Omorfi produces analyses in its own format, which has some degree of compatibility with tags from the Universal Dependencies (UD) project (Pyysalo et al., 2015). As an example, kakusta may be analysed as `[WORD_ID=kakku][UPOS=NOUN][NUM=SG][CASE=ELA]`. Morphological tags are mapped to analytical morphemes so that e.g. `[CASE=ELA]` is output as -sta, while `WORD_ID` is passed through. The order in which the tags appear is the same order as the surface morphemes appear, meaning our analytical morphemes are in the same order as the surface morphemes.

Wiktionary contains various template tags which give information about word formation. This data is scraped into a database so that each etymology section can give a derivation for its headword. Template tags are normalised into either inflections, derivations or compounds consisting of normalised segments. For compounds and most derivations, normalised segments are directly available as arguments to the template tag. However, the `agent noun of` template tag, for example, must be manually mapped to "-ja". Finally, the `form of` template tag makes use of grammatical terms such as elative, which are mapped to normalised segments such as "-sta".

4.3 Building segmentation derivations

Complex words may have several levels of compounding or word derivation and inflection. Thus, we may have to make use of several lookups to fully segment a word form. We also want to make sure a completely segmented word form can be associated with all lexical items that make it up. We thus shift our perspective to think of these analyses as rules and the segmenter as a rule engine which applies them to produce derivations subject to constraints. Each rule can match any single segment and produce many segments.

The basic rule engine operates by recursively applying rules. It keeps track of the current front of the derivation tree. At each iteration, each node from the front is considered and one or more steps consisting of applying one or more rules are taken to create child nodes, creating a new front. There may be multiple rules which can match a segment. In this case, all combinations of rules matching each matchable segment are applied. When either there are no more rules which match, or there is a match which does not expand any segments, the node is marked as terminal.

A simple approach would be to allow all rules to apply at once. However, Omorfi analyses do not work very well as rules as-is in our case, for example for voileipäkakusta Omorfi produces three different analyses of different levels of decompounding of voileipäkakku. If we were to apply each of these analyses as rules we would end up with 3 final segmentations. However, for our purposes, they should all be subsumed under the same derivation. Therefore we take the following approach:

[6] The MWE extraction code is made available at https://github.com/frankier/lextract

1. First, apply Wiktionary based rules recursively.
2. Fetch all Omorfi rules resulting from looking up the whole word form
3. While there are Omorfi rules left:
   (a) Remove any Omorfi rules already subsumed by the current derivation.
   (b) If any rules remain, apply the one producing the least new segments and discard.
   (c) Apply Wiktionary based rules recursively.
4. Apply any retrofitting rules, which exist to deal with occasional cases of fusional Finnish morphology.

For example voileipäkakusta (English: out of/from sandwich cake) would produce the following derivation:

Example 1: voileipäkakusta

→ voileipäkakku sta	*Omorfi: voileipäkakusta*
→ voileipä kakku sta	*Wiktionary: voileipäkakakku*
→ voi leipä kakku sta	*Wiktionary: voileipä*

While voimakkaammin (English: more powerfully) would produce the following:

Example 2: voimakkaammin

→ voimakkaasti mpi	*Wiktionary: voimakkaammin*
→ voimakas sti mpi	*Wiktionary: voimakkaasti*
→ voima kas sti mpi	*Wiktionary: voimakas*
→ voida ma kas sti mpi	*Wiktionary: voima*

In this case a retrofitting rule mmin → sti mpi[7] can be applied:

Example 3: voimakas sti mpi

← voimakas mmin	*Retrofit: mmin*
← voimakkaammin	*Connect to parent*

4.4 Constraints upon rules

Applied as-is, this scheme will produce impossible segmentations. However, if we consider the POS (Part-Of-Speech) of each segment, we can place constraints to ameliorate this.

We use a simple set of POS tags based on WordNet: Verb, Noun, Adverb, Adjective & Unknown. The UD POS tags used by Omorfi and the Wiktionary POS headings are mapped into this common scheme. The mapping is lossy, for example, UD adpositions are mapped onto the Adverb POS. All closed classes, interjections and affixes are mapped to Unknown. Note that constituent words of Finnish compounds can be inflected words, and so here inflected forms are treated as having the POS of their lemma.

[7] "-sti" is adverb forming morpheme like "-ly", while "-mpi" is a comparative forming morpheme like "-er".

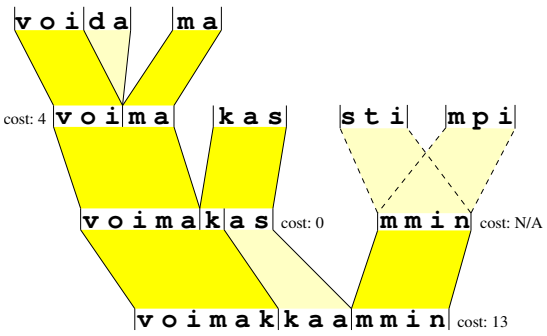

Figure 3: Alignment within derivation tree of voimakkaammin. Dark yellow portions denote surface spans, while each of the whole yellow portions including dark and light denote the whole logical span. The cost of each alignment according to Force-Align is shown next to the parent segment. The dashed lines indicate the alignment is not produced by Force-Align but instead obtained from the underlying rule. In this case: the synthetic rule -mmin → -sti -mpi.

The permissible compound POS patterns can then be produced by a list of production rules, obtained by studying Hyvärinen (2019):

Verb → *Noun Verb* (e.g., koe+lentää)
Verb → *Adverb Verb* (e.g., edes+auttaa)
Noun → *Noun Noun* (e.g., voi+leipä)
Noun → *Adjective Noun* (e.g., puna+viini)
Adjective → *Adjective Adjective* (e.g., hyvän+näköinen)

We start by treating the whole token as having Unknown POS, meaning we can match any POS. At any time a segment is constrained to having one of a set of POS tags. For compounds of more than two parts, we can obtain the possible POS patterns by expanding the production rules given above.

Referring back to Example 1, a Wiktionary rule allows the analysis of voi (Verb) as voida (3rd pers.), however voileipäkakku is known to be a Noun, meaning according to the above rules, voi must be either a Noun or an Adjective, meaning this rule cannot be applied.

4.5 Obtaining alignments from derivations

In order to find correspondences between individual characters in analytical segments, surface forms and headwords, we apply the Force-Align procedure at each step of the analytical segmentation derivation. Each child segment is given a span into the parent segment. These are ordered and non-overlapping. Matches are performed af-

Function FORCE-ALIGN(parent string p, array of child strings $c_1 \ldots c_n$)
returns alignment a
  Create a bounded priority queue pq with the lowest cost partial solution at its front
  Add an empty partial solution into pq
  while the head of pq is not complete **do**
    Pop partial solution j from front of pq
    /* Make a match */
    if j's cursor into c has not reached end **and** j's cursor into p has not reached end **and** $p_{(j\text{'s cursor into } p)} = c_{(j\text{'s cursor into } c)}$ **then**
      Add copy of j into pq with its cursors into p and c incremented
    end
    /* Skip a parent character */
    if j's cursor into p is not at beginning or end **then**
      Add copy of j into pq with its cursor into p and its *parent characters skipped* incremented
    end
    /* Skip the rest of the current child segment */
    if j's cursor into p is not at beginning **and** j's cursor into c has not reached end **then**
      Add copy of j into pq with its cursor into c and its *child characters skipped* incremented
    end
  end
  $a :=$ alignment formed by solution at front of pq
end

Algorithm 1: The Force-Align procedure to find an alignment between a parent string and its segmented children strings.

ter normalisation. All strings are lowercased and the front vowels ä, ö and y are mapped to the respective back vowels a, o and u. We aim to minimise a cost defined as the sum of the square of the number of parent characters skipped and square of the number of child characters skipped. An example showing the type of alignments produced by Force-Align applied to the derivation of voimakkaammin is shown in Figure 3.

Force-Align is implemented as a dynamic programming style procedure, given as pseudocode in Algorithm 1. At each step, Force-Align keeps track of candidate solutions in a priority queue, with the lowest cost partial solution always being at the front. The priority queue has bounded length to bound the running time — making the procedure a form of beam search. Whenever a partial candidate solution is taken from the front of the queue, up to three new partial solutions are created and added back to the priority queue: making a single character match; skipping a single character from the parent string; and skipping the rest of the characters in the current child segment. The procedure ends when there is a complete solution at the front of the queue. Each child segment's full span covers the characters from the beginning of its first character match to just before the first character match of the next child segment, or until the end of the parent segment in the case of the last child segment.

A span of a segment onto any ancestor segment can be found by following a simple rule at each step: shift the whole span rightwards by far enough to fit any new segments to the left, and extend the right edge to the end of the child span onto the parent segment while the current child segment is the rightmost. As an example, consider Figure 3 and the analytical morpheme -kas. It begins on the right edge. We consider its alignment onto voimakas and find we must shift its left edge by five characters to make space for voima. We replace its right edge with that of its parent, which does not change the span. -kas is still on the right edge of voimakas when we consider the alignment of voimakas and voimakkaammin. At this step, we do not have to shift the left edge since there are no new segments to the left. The right edge is replaced with the right edge of the alignment of voimakas onto voimakkaammin, shifting it right by one character. The final span contains the characters 'kkaa' — which is the allomorph corresponding to 'kas', as required.

When the user hovers over a segment of a segmentation in the user interface, the corresponding surface form of the same segment should highlight. The highlighting consists of a strong highlight for that part of the surface form which has overlapping text with the analytic form, and a weak highlight for that part which is grammatically part of the same morpheme but does not literally match. The strong highlight is found by finding the longest match between the child segment and its span within the parent segment, while the weak highlight is made from any part of the span which is left over.

Special consideration is given to wildcards, such as ___-sta. In this case, matching is performed right to left, and the wildcard is weakly matched against that which remains after all other analytic segments are aligned.

Figure 4: A screenshot showing the Finnish Wikipedia page *Turku* being read using the browser extension.

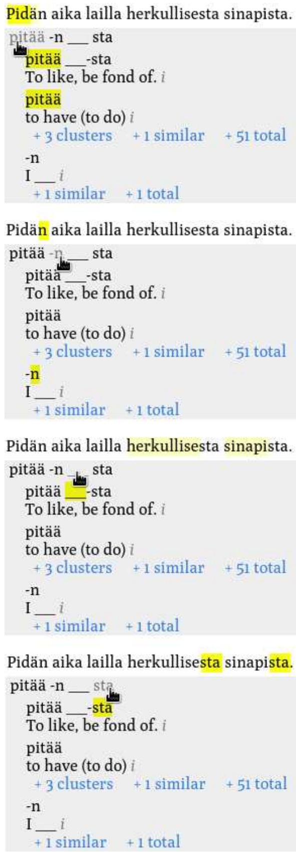

Figure 5: A composite screenshot showing different stages of the interaction resulting when a user brushes over segments in the text analyser.

5 Visualising Finnish word formation

A screenshot of the user interface is shown in Figure 4. Definitions are grouped by normalised segmentation. Within each normalised segmentation there are defined headwords, each corresponding to one or more of the normalised segments. They are ordered in decreasing order of coverage of the normalised segmentation, meaning those definitions which define the meaning of the surface form most closely appear closest to the top. Within each defined headword appears one or more clusters of definitions, each with an exemplar.

To bring attention back to surface forms from the normalised forms, the interface highlights the surface forms as the learner hovers over the segmented forms, as shown in Figure 5. The interaction recalls a one dimensional "hover scrub" action. Initially, the whole word or phrase is lightly highlighted. As the learner scrubs over analytic morphemes, the corresponding spans in the surface form are highlighted.

To show the connection between the normalised segmentation and its definitions, parts of the defined headwords are highlighted when normalised segments are hovered over, as shown in Figures 4 & 5. The whole interaction serves to link the different views of surface form, analytical form and headwords.[8]

6 Conclusions and Future Work

This paper presented *NiinMikäOli?!* The system streamlines the experience of using reference material by presenting it in-context, emphasising the most important parts and presenting simplified grammatical analyses which do not rely upon technical linguistic jargon, following the principle of "show, don't tell".

Clearly, the question of whether systems such as *NiinMikäOli?!* truly help language learners is a pertinent one. Quantitative user evaluation to validate existing features and point to new ones is thus an important piece of future work.

NiinMikäOli?! gives definitions in English. Adding common L1 languages of Finnish learners such as Swedish, Russian or Arabic, as well as Finnish itself could be a useful addition.

The current analytical segmenter is rule based, and thus cannot handle out of vocabulary words. A machine learning approach such as that of Kann et al. (2016) could be combined with the data developed here to address this.

A future direction for all reading assistants is better prediction of language learner needs, which would lead to a system which knows beforehand which types of reading assistance would be best to offer either by explicitly requesting information from the learner, or implicitly using information from previous interactions with the software.

[8] The NiinMikäOli?! browser extension and website are available at https://niinmikaoli.fi/, while the analytical segmentation code is available at https://github.com/frankier/asafi.

References

Eneko Agirre, Oier Lopez de Lacalle, and Aitor Soroa. 2014. Random walks for knowledge-based word sense disambiguation. *Computational Linguistics*, 40(1):57–84.

Werner Bleyhl. 2009. The hidden paradox of foreign language instruction. or: Which are the real foreign language learning processes. *Input Matters in SLA. Clevedon: Multilingual Matters*, pages 137–55.

Brendan J. Frey and Delbert Dueck. 2007. Clustering by passing messages between data points. *Science*, 315(5814):972–976.

Robert Godwin-Jones. 2018. Contextualized vocabulary learning. *Language Learning & Technology*, 22(3):1–19.

Irma Hyvärinen. 2019. *Compounds and multi-word expressions in Finnish: Compounds and Multi-Word Expressions*, pages 307–336. De Gruyter.

Jenna Kanerva, Filip Ginter, Niko Miekka, Akseli Leino, and Tapio Salakoski. 2018. Turku neural parser pipeline: An end-to-end system for the conll 2018 shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 133–142. Association for Computational Linguistics.

Katharina Kann, Ryan Cotterell, and Hinrich Schütze. 2016. Neural morphological analysis: Encoding-decoding canonical segments. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 961–967, Austin, Texas. Association for Computational Linguistics.

Fred Karlsson. 2015. *Finnish: an essential grammar*. Routledge.

Anisia Katinskaia, Javad Nouri, and Roman Yangarber. 2017. Revita: a system for language learning and supporting endangered languages. In *Proceedings of the joint workshop on NLP for Computer Assisted Language Learning and NLP for Language Acquisition*, pages 27–35, Gothenburg, Sweden. LiU Electronic Press.

Maddalen Lopez de Lacalle, Egoitz Laparra, Itziar Aldabe, and German Rigau. 2016. A multilingual predicate matrix. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 2662–2668, Portorož, Slovenia. European Language Resources Association (ELRA).

Veronika Laippala and Filip Ginter. 2014. Syntactic n-gram collection from a large-scale corpus of internet finnish. In *Human Language Technologies-The Baltic Perspective: Proceedings of the Sixth International Conference Baltic HLT*, volume 268, page 184.

P. M. Lightbown and N. Spada. 2013. *How languages are learned*, 3rd ed edition. Oxford handbooks for language teachers. Oxford University Press.

Krister Lindén and Lauri Carlson. 2010. Finnwordnet–finnish wordnet by translation. *LexicoNordica–Nordic Journal of Lexicography*, 17:119–140.

Detmar Meurers, Ramon Ziai, Luiz Amaral, Adriane Boyd, Aleksandar Dimitrov, Vanessa Metcalf, and Niels Ott. 2010a. Enhancing authentic web pages for language learners. In *Proceedings of the NAACL HLT 2010 Fifth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 10–18. Association for Computational Linguistics.

Detmar Meurers, Ramon Ziai, Luiz Amaral, Adriane Boyd, Aleksandar Dimitrov, Vanessa Metcalf, and Niels Ott. 2010b. Enhancing authentic web pages for language learners. In *Proceedings of the NAACL HLT 2010 Fifth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 10–18. Association for Computational Linguistics.

J. Nerbonne, D. Dokter, and P. Smit. 1998. Morphological processing and computer-assisted language learning. *Computer Assisted Language Learning*, 11(5):543–559.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830.

Tommi A Pirinen. 2015. Development and use of computational morphology of finnish in the open source and open science era: Notes on experiences with omorfi development. *SKY Journal of Linguistics*, 28:381–393.

Sampo Pyysalo, Jenna Kanerva, Anna Missilä, Veronika Laippala, and Filip Ginter. 2015. Universal dependencies for Finnish. In *Proceedings of the 20th Nordic Conference of Computational Linguistics (NODALIDA 2015)*, pages 163–172, Vilnius, Lithuania. Linköping University Electronic Press, Sweden.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Robert Reynolds, Eduard Schaf, and Detmar Meurers. 2014. A VIEW of Russian: Visual input enhancement and adaptive feedback. In *Proceedings of the third workshop on NLP for computer-assisted language learning*, pages 98–112, Uppsala, Sweden. LiU Electronic Press.

Frankie Robertson. 2020. Filling the ___-s in finnish mwe lexicons. In *Joint Workshop on Multiword Expressions and Electronic Lexicons (MWE-LEX 2020)*, Online. Association for Computational Linguistics.

Richard W. Schmidt. 1990. The Role of Consciousness in Second Language Learning. *Applied Linguistics*, 11(2):129–158.

Robyn Speer, Joshua Chin, Andrew Lin, Sara Jewett, and Lance Nathan. 2018. Luminosoinsight/wordfreq: v2.2.

Leonardo Zilio, Rodrigo Wilkens, and Cédrick Fairon. 2017. Using NLP for enhancing second language acquisition. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 839–846, Varna, Bulgaria. INCOMA Ltd.