# Tools for The Production of Analogical Grids and a Resource of N-gram Analogical Grids in 11 Languages

**Rashel Fam, Yves Lepage**
Graduate School of Information, Production, and Systems
Waseda University
2-7 Hibikino, Wakamatsu-ku, Kitakyushu-shi, 808-0135 Fukuoka-ken, Japan
{fam.rashel@fuji., yves.lepage@}waseda.jp

**Abstract**

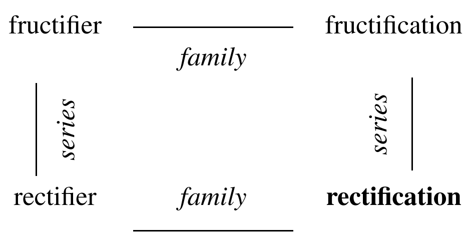
We release a Python module containing several tools to build analogical grids from words contained in a corpus. The module implements several previously presented algorithms. The tools are language-independent. This permits their use with any language and any writing system. We hope that the tools will ease research in morphology by allowing researchers to automatically obtain structured representations of the vocabulary contained in corpora or linguistic data. We also release analogical grids built on the vocabularies contained in 1,000 corresponding lines of the 11 different language versions of the Europarl corpus v.3. The grids were built on N-grams of different lengths, from words to 6-grams. We hope that the use of structured parallel data will foster research in comparative linguistics.

**Keywords:** Analogy, morphology, analogical grids

## 1. Introduction

*Paradigm tables* are known for their usefulness in learning conjugation or declension when studying a language. Such paradigm tables are the result of grammatical tradition or thorough linguistic formalization. They are commonly found in dictionaries and constructed from lexemes and exponents, like the one shown in Figure 2 (*left*) which is taken from a French & English dictionary (Mansion, 1981, grey section, p. 1).

*Analogical grids* are not paradigm tables, but they also give a compact view on the organization of a lexicon up to a certain extent (see below, Section 2.1.). Analogical grids are the result of an empirical procedure. They may be seen as a preliminary step towards the production of paradigm tables. Figure 2 (*right*) shows an example of an analogical grid in English. They can be used to study the productivity of a language (Singh and Ford, 2000; Neuvel and Fulop, 2002; Hathout, 2008). Fam and Lepage (2016) performed such an analysis across 12 languages using analogical grids built from the Bible corpus (Christodouloupoulos, 2015). As another example of use, Hathout (2009) showed how to produce the French word form *rectification* by analogy from the neighboring word forms *fructifier*, *fructification*, and *rectifier* in the same series (see Figure 1).



*fructifier* : *fructification* :: *rectifier* : *rectification*

Figure 1: Producing a new word form from neighboring word forms. Example taken from (Hathout, 2009)

This paper introduces the release of a Python module which implements previously presented algorithms that automatically build analogical grids. We also release analogical clusters and analogical grids produced from a parallel corpus on 11 European languages using this Python module.

## 2. Main Usages of the Tools Released

We release an implementation of previously presented algorithms to produce analogical grids as a Python 2 module[1] called *Nlg*. The various algorithms have been presented elsewhere (Lepage, 1998; Lepage, 2014; Fam and Lepage, 2016). One particular program called *Words2Grids* in this module simply takes a list of word forms as input and delivers a list of analogical grids. Each word form in the list is converted into a feature vector before analogical grids are constructed from such feature vectors. The module also provides another program, *Words2Vectors*, to produce feature vector representations either directly from word forms or from descriptions of word forms. The following sections introduce several ways to use the Python module.

### 2.1. From Word Forms to Analogical Grids

The default usage of the module is to produce analogical grids from a list of word forms. An analogical grid is a matrix of words where four words from two rows and two columns are a proportional analogy. Formula (1) gives the definition of an analogical grid.

$$\begin{array}{cccc} P_1^1 : & P_1^2 : & \cdots : & P_1^m \\ P_2^1 : & P_2^2 : & \cdots : & P_2^m \\ \vdots & \vdots & & \vdots \\ P_n^1 : & P_n^2 : & \cdots : & P_n^m \end{array} \quad \overset{\Delta}{\Longleftrightarrow} \quad \begin{array}{c} \forall (i,k) \in \{1,\ldots,n\}^2, \\ \forall (j,l) \in \{1,\ldots,m\}^2, \\ P_i^j : P_i^l :: P_k^j : P_k^l \end{array} \tag{1}$$

Analogy is defined from feature vectors representing word forms, through equality of ratios. A ratio is the difference between two feature vectors plus the edit distance between

[1] `lepage-lab.ips.waseda.ac.jp/nlg-module`

|  | *Infinitive* | *Preterit* | *Past participle* | *Present participle* |
|---|---|---|---|---|
| *Regular* | walk | walked | walked | walking |
| *verb* | smoke | smoked | smoked | smoking |
| *Irregular* | write | wrote | written | writing |
| *verb* | think | thought | thought | thinking |

|  |  |  |  |
|---|---|---|---|
| walk : | walks : | walking : | walked |
| show : | shows : | showing : | showed |
| open : | opens : | opening : |  |
| study : | : | studying : |  |
| play : | : | playing : | played |

Figure 2: A paradigm table (*left*) taken from a French & English dictionary (Mansion, 1981) and an analogical grid (*right*) obtained by our tools (and reduced to a few rows for lack of space)

the word forms. We refer the reader to (Fam and Lepage, 2016) for exact definitions.
In this setting, a word form is represented as a vector of features which are simply the number of occurrences for all the characters in the alphabet. For instance, in lowercase English, the dimension of the vector is 26 (from *a* to *z*) as illustrated in Formula (2). Here, the notation $|A|_c$ stands for the number of occurrences of character $c$ in string $A$.

$$A = \begin{pmatrix} |A|_a \\ |A|_b \\ \vdots \\ |A|_z \end{pmatrix} \qquad \text{walking} = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \tag{2}$$

The right part of Figure 2 shows few lines of an analogical grid that has been obtained on a list of English word forms with the program *Words2Grids*.

## 2.2. From Morphological Features to Paradigm tables

The previous use of the released tools automatically converts word forms into specific feature vectors. In opposition to that, it is possible for the user to produce real paradigm tables from feature vectors standing for actual morphological features, like lemma, part-of-speech, case, tense. Such feature vectors can be built, for instance, from the Unimorph Project (Kirov et al., 2016) data which have been built from parsing Wiktionary data into a language-independent feature schema (Sylak-Glassman et al., 2015b; Sylak-Glassman et al., 2015a). Formula (3) illustrates the representation of the word form *walking*: its lemma is *to walk* and it has the verb (VB), present (PRST), participle (PTCP) tags as morphological features. For the purpose of inner processing, the labels are converted into Boolean values.

$$A = \begin{pmatrix} lemma = \text{"to walk"}(A) \\ is\_VB(A) \\ is\_NN(A) \\ is\_PRST(A) \\ \vdots \\ is\_PTCP(A) \end{pmatrix} \qquad \text{walking} = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \\ \vdots \\ 1 \end{pmatrix} \tag{3}$$

Figure 3 shows an example of a paradigm table built from a list of word forms described by morphological features. For some lemmas, some cells may be empty. These lemmas are not necessarily defective; this simply means that the forms did not appear in the input data. It should be stressed that the names of the morphological features are not shown in the paradigm tables output by our programs.

|  |  |  |
|---|---|---|
| initiate : | initiated : | initiates |
| undercry : | : | undercries |
| mummify : | mummified : |  |
| tole : | tollen : |  |

Figure 3: A paradigm table built from a list of word forms annotated with morphological features in English

## 2.3. From User-Defined Features to New Types of Grids

It is possible to directly use user-defined features, i.e., richer vector representations of word forms as input to the released programs. As an example, the feature vector shown in Formula (4) concatenates the two types of vectors presented in the previous sections.

$$A = \begin{pmatrix} lemma = \text{"to walk"}(A) \\ |A|_a \\ \vdots \\ |A|_z \\ is\_VB(A) \\ is\_NN(A) \\ is\_PRST(A) \\ \vdots \\ is\_PTCP(A) \end{pmatrix} \qquad \text{walking} = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 0 \\ 1 \\ 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix} \tag{4}$$

From such feature vectors, the programs output regular paradigm tables (see Figure 4 as an example). We hope that researchers will freely define their own types of feature vector representations to build analogical grids that correspond to their own needs.

|  |  |  |
|---|---|---|
| initiate : | initiated : | initiates |
| stage : | staged : |  |
| elucidate : | elucidated : |  |
| assume : | assumed : |  |

Figure 4: A regular paradigm table built from combining two previous feature vectors

## 2.4. An Example of How to Use the Tools

In this section, we show an example of how to use the tools to produce analogical grids from a text. The following illustration is taken from (Fam and Lepage, 2017).
Consider that we have a text as shown in the top of Figure 6. It is a forged example in Indonesian, a language known for its richness of derivational morphology. Using a tokenizer, we can get a list of tokens and then obtain a set of words

**RAW output:** *anto.grid.txt*

```
 minum :  meminum :  diminum :  minuman ::  makan :  memakan :  dimakan :  makanan ::
main :  None :  None :  mainan ::  beli :  None :  dibeli :  None
```

**Pretty_print output:** *anto.prettygrid.txt*

```
# Grid no.:  1 - Attributes(length=4, width=4, size=16, filled=12, saturation=0.75)
minum : meminum : diminum : minuman
makan : memakan : dimakan : makanan
main  :         :         : mainan
beli  :         : dibeli  :
```

Figure 5: Analogical grids in Indonesian extracted from a set of word in the bottom of Figure 6. It is shown as RAW output (*top*) and printed using **pretty_print** option (*bottom*).

*Anto memakan nasi dan meminum air. Nasi itu dibeli di pasar. Di pasar, Anto melihat mainan. Anto senang main bola. Setelah main, Anto suka minum es dan makan cilok. Makanan dan minuman itu juga dia beli di pasar. Es dan cilok memang enak dimakan dan diminum selesai olahraga.*

*air anto* ***beli*** *bola cilok dan di dia* ***dibeli dimakan diminum*** *enak es itu juga* ***main mainan makan makanan*** *melihat* ***memakan*** *memang* ***meminum minum minuman*** *nasi olahraga pasar selesai senang setelah suka*

Figure 6: A text in Indonesian (*top*) and the list of words extracted from it (*bottom*). Words appearing in Figure 5 are boldfaced.

(types) from it as shown in the bottom part of Figure 6. Let us say that we prepare a file named *anto.words.txt* which contains all of the types, one type per line. We can then extract the analogical grids by running the following command.

```
 $ python Words2Grids.py
< anto.words.txt
> anto.grid.txt
```

The output of the command is printed into the file named *anto.grid.txt*. The content of the file is shown at the top of Figure 5. For the sake of development of the module (input-output between modules), we choose to encapsulate the analogical grids in such data format defined in Section 3.1. Caution: *None* stands for an empty cell.

### 2.4.1. Visualizing the Analogical Grids

To have a better view on the analogical grids produced by the tools, we provide an option called **pretty_print**. Running the following command will print the analogical grids in a separate file named *anto.prettygrid.txt* with a different format than the one mentioned in Section 3.1.

```
 $ python Words2Grids.py
< anto.words.id.txt
--pretty-print anto.prettygrid.txt
```

The attributes of the grids are printed on top of each grid.

- **length**: number of rows
- **width**: number of columns
- **size**: size of the analogical grid
- **filled**: number of non-empty cells
- **saturation**: saturation of the analogical grid

The bottom of Figure 5 shows the analogical grid extracted from the set of words in Figure 6 printed using the **pretty_print** option. It has a size of 16 cells with 4 rows and 4 columns. 12 cells out of 16 cells are filled so that the saturation of the grid is 0.75. Further details about the attributes of analogical grids, like size and saturation, will be described in Section 3.3. and Section 3.4.

### 2.4.2. Extracting Analogical Grids Around Particular Word Forms

We also offer a function to focus the study on one particular word form: it is possible to deliver only those analogical grids which contain the particular word form under scrutiny.
The *Words2Grids* program runs in fact in two steps. It first extracts analogical clusters from a list of word forms and then builds analogical grids from the extracted clusters. First, we extract all analogical clusters using the *Words2Clusters* program with a specific option called *focus*. This option will only extract analogical clusters which contain a particular word given as parameter of the option. We then use the *Clusters2Grids* program to produce the analogical grids from the previously extracted clusters.

However, all of these procedures are performed seamlessly to the users by the *Words2Grids* program. For example, to produce all analogical grids which contain the word *walking*, we can run the following command.

```
 $ python Words2Grids.py
< anto.words.id.txt
--focus 'walking'
```

By building analogical grids from those clusters, the user obtains all the analogical grids which contain that word form. It is then possible to characterize the actual productivity of a particular word form by inspecting the size of the analogical grids that contain it. For further study, as proposed in (Hathout, 2009), one can then retrieve all the word forms that have the same relationship with the word form under scrutiny. These words are basically the neighboring word forms inside the analogical grids.

## 3. Languages and Data in the Released Resource

By using the procedure described in Section 2.1., we produced analogical grids from all the words contained in 1,000 corresponding lines of the Europarl corpus v3 (Koehn, 2007) in all of the 11 European languages. The motivation for using a multilingual parallel corpus is to base on the analogical grids produced to perform comparative studies across these languages. We produced the analogical grids for different sizes of N-grams, from unigrams to six-grams.

| Language | # tokens ($N$) | # types ($V$) | Avg length of types |
|---|---|---|---|
| da | 27,034 | 5,304 | 9.06±4.22 |
| de | 27,042 | 5,753 | 9.69±4.19 |
| el | 28,559 | 6,397 | 16.45±6.21 |
| en | 28,594 | 4,305 | 7.35±2.77 |
| es | 29,974 | 5,300 | 8.18±2.91 |
| fi | 20,604 | 7,473 | 10.60±4.22 |
| fr | 31,257 | 5,184 | 8.25±3.01 |
| it | 28,269 | 5,425 | 8.08±2.84 |
| nl | 28,933 | 5,028 | 8.90±3.99 |
| pt | 29,342 | 5,472 | 8.32±3.04 |
| sv | 25,681 | 5,452 | 9.03±4.11 |

Table 1: Statistics on the first 1,000 lines of Europarl corpus

Table 1 shows the statistics of the input data. French has the largest number of tokens with more than thirty thousand tokens. Although Finnish has the smallest number of tokens with around twenty thousand tokens, it has the largest number of types followed by Greek. It is due to the characteristic of Finnish being an agglutinative language. This is also reflected by the average length of types. Finnish has the second longest average length (around 11 characters per type in average) of types after Greek (around 17 characters per type in average), contrary to the order of largest number of types. The other languages tend to have around twenty eight thousand tokens represented by around five thousand types with average length of 9 characters.

### 3.1. Data Format

We release a complete dataset[2] of analogical clusters and analogical grids produced on the data described in the previous section. Each file contains a list of analogical clusters or analogical grids.
As for the analogical clusters, each line in a file is one analogical cluster, i.e., a list of ratios separated by two colons. A ratio consists in two strings separated by a colon. The format of is thus as follows.

$$\mathtt{A}_1 : \mathtt{B}_1 :: \mathtt{A}_2 : \mathtt{B}_2 :: \mathtt{A}_3 : \mathtt{B}_3 :: \ \ldots \ :: \mathtt{A}_N : \mathtt{B}_N$$

As for analogical grids, each line in a file is one analogical grid, i.e., a list of rows separated by two colons. A row is a fixed list of strings separated by colons. The format of is as follows.

$$\mathtt{W}_1^1 : \mathtt{W}_1^2 : \ \ldots \ \mathtt{W}_1^m :: \mathtt{W}_2^1 : \mathtt{W}_2^2 : \ \ldots \ : \mathtt{W}_2^m :: \ \ldots \ \mathtt{W}_n^m$$

Indeed, an analogical cluster is just an analogical grid with 2 columns and no empty cell.

### 3.2. Number of Analogical Grids Produced

Figure 7 (*top*) shows the number of analogical grids produced for each language with different sizes of N-grams (caution: scale in ten thousands of grids). The number of analogical grids produced rises from unigrams to bigrams, and decreases afterwards in all languages, except Finnish and Greek.
For Finnish, an explanation may be that Finnish, being an agglutinative language, a single word form in that language corresponds in fact to several words in the other European languages, i.e., unigrams in Finnish correspond to bigrams or trigrams in the other European languages. Finnish also exhibits the highest number of analogical grids output for unigrams. This reflects the morphological richness of this language in comparison to the other languages.
For Greek, the number of analogical grids is stable at around 1,000 analogical grids for all sizes if N-grams. The number of analogical grids seems to be stable across N-grams.
The average size of analogical grids exhibit high fluctuations from unigrams to fourgrams. A drop is observed afterwards. The difference is up to five times for languages like English, Spanish, French, and Italian. However, we do not see that for Greek.

### 3.3. Average Size of Analogical Grids

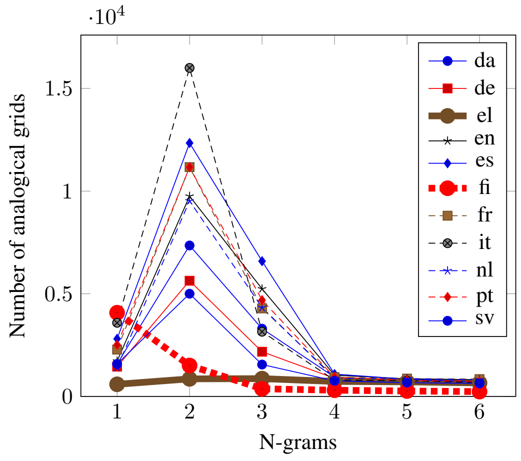
The size of an analogical grid is defined as its number of rows multiplied by its number of columns. (See Formula (5)). E.g., the analogical grid in Figure 2 has a size of $5 \times 4 = 20$.

$$\text{Size} = \text{Number of rows} \times \text{Number of columns} \qquad (5)$$

[2]`lepage-lab.ips.waseda.ac.jp/nlg-grids-dataset`

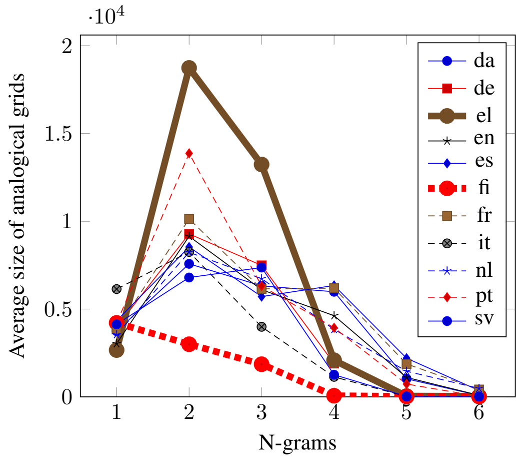
**Number of analogical grids produced**

| Language | N-grams | | | | | |
|---|---|---|---|---|---|---|
| | N = 1 | N = 2 | N = 3 | N = 4 | N = 5 | N = 6 |
| da | 1,581 | 7,352 | 3,299 | 942 | 679 | 673 |
| de | 1,451 | 5,634 | 2,177 | 884 | 652 | 669 |
| el | 591 | 856 | 867 | 731 | 710 | 657 |
| en | 1,723 | 9,754 | 5,239 | 1,034 | 860 | 813 |
| es | 2,806 | 12,342 | 6,587 | 1,088 | 840 | 708 |
| fi | 4,077 | 1,506 | 373 | 303 | 264 | 235 |
| fr | 2,282 | 11,170 | 4,287 | 940 | 870 | 844 |
| it | 3,586 | 15,999 | 3,152 | 791 | 767 | 676 |
| nl | 1,408 | 9,522 | 4,331 | 879 | 861 | 812 |
| pt | 2,514 | 11,164 | 4,690 | 940 | 785 | 734 |
| sv | 1,579 | 5,002 | 1,553 | 759 | 672 | 633 |



**Average size of analogical grids**

| Language | N-gram | | | | | |
|---|---|---|---|---|---|---|
| | N = 1 | N = 2 | N = 3 | N = 4 | N = 5 | N = 6 |
| da | 3735 | 7576 | 6320 | 5985 | 1025 | 76 |
| de | 3967 | 9284 | 7486 | 1896 | 38 | 18 |
| el | 2663 | 18738 | 13237 | 2081 | 40 | 29 |
| en | 3033 | 9143 | 6102 | 4622 | 1104 | 85 |
| es | 4333 | 8534 | 5709 | 6343 | 2199 | 359 |
| fi | 4212 | 2994 | 1855 | 61 | 18 | 15 |
| fr | 3836 | 10128 | 6149 | 6188 | 1873 | 413 |
| it | 6137 | 8252 | 3988 | 1136 | 16 | 14 |
| nl | 3543 | 8234 | 6732 | 3843 | 1457 | 480 |
| pt | 4203 | 13870 | 6314 | 3934 | 728 | 22 |
| sv | 4126 | 6796 | 7365 | 1258 | 19 | 18 |



**Average saturation of analogical grids**

| Language | N-gram | | | | | |
|---|---|---|---|---|---|---|
| | N = 1 | N = 2 | N = 3 | N = 4 | N = 5 | N = 6 |
| da | 93.71 | 95.20 | 96.01 | 97.21 | 98.58 | 98.64 |
| de | 93.77 | 95.10 | 95.35 | 97.94 | 98.44 | 98.45 |
| el | 94.48 | 95.13 | 95.74 | 97.88 | 98.14 | 98.88 |
| en | 93.79 | 95.63 | 96.06 | 97.68 | 99.03 | 98.19 |
| es | 93.78 | 95.45 | 96.55 | 97.35 | 98.94 | 98.13 |
| fi | 94.31 | 96.12 | 96.30 | 97.29 | 98.58 | 98.43 |
| fr | 94.19 | 95.57 | 96.78 | 97.72 | 98.94 | 98.47 |
| it | 93.57 | 95.55 | 97.37 | 98.80 | 98.92 | 99.40 |
| nl | 93.61 | 95.29 | 96.52 | 98.29 | 98.88 | 99.10 |
| pt | 93.89 | 95.29 | 96.42 | 98.16 | 99.01 | 98.72 |
| sv | 93.45 | 95.10 | 95.94 | 97.92 | 98.76 | 99.27 |

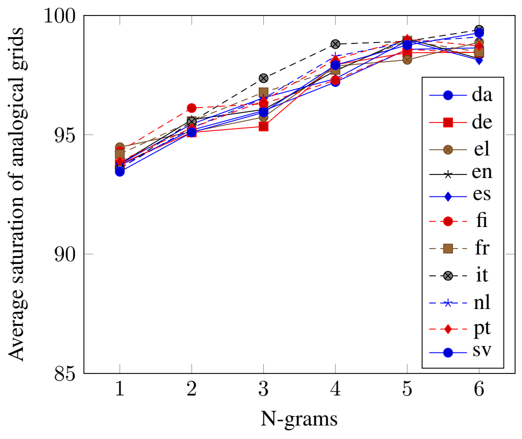


Figure 7: Number (*top*), average size (*middle*), and average saturation (*bottom*) of analogical grids produced for the first 1,000 lines of Europarl with different size of N-grams.

As can be seen from Figure 7, we observe a similar behaviour for average size and number of analogical grids produced. The average size of the analogical grids in all languages reaches its peak on bigrams and then goes down until sixgrams, except for Swedish. The same interesting phenomenon in Finnish occurs again as it has lower average size for longer N-gram. On the other hand, Greek has the lqrgest average size for bigrams and trigrams despite having the smallest average size for unigrams. This might be caused by Greek having a richer system of auxiliary verbs. Said in another way, one word in other languages may be expressed with two or more words in Greek. Swedish is another anomaly: the largest average size for Swedish is observed on trigrams instead of bigrams.

### 3.4. Saturation of Analogical Grids

The number of empty cells of an analogical grid can be roughly considered the number of possible forms not present in the data. We call *saturation*[3] the ratio of the number of non-empty cells and the size of an analogical grid in percentage. E.g., the analogical grid in Figure 2 has 4 empty cells; its saturation is thus: $(1.0-4/(4\times 5))\times 100 = 80\,\%$.

$$\text{Saturation} = 100 - \frac{\text{Number of empty cells } \times 100}{\text{Total number of cells}} \qquad (6)$$

The average saturation of analogical grids produced in 11 languages of our Europarl data tends to rise from unigrams to sixgrams (see Figure 7 (*bottom*)). This indicates that analogical grids on longer N-grams are more dense. However, it should be kept in mind, by referring to the results on number of analogical grids and their average size, that they are much less numerous and much smaller.

Empty cells can be filled by potential word forms. Thus, they illustrate productivity in language. The confidence with which empty cells can be filled is of course linked with the problem of over-generation by analogy. These potential word forms are basically unseen from the corpus used to build the analogical grids. Studying the productivity of correct word forms from analogical grids would be another intriguing experiment to perform. We hope that such a resource can be used by other researchers for re-inflection tasks or to address to some extent the out-of-vocabulary (OOV) problem.

## 4. Conclusion

We released a Python module for the production of analogical grids from word forms contained in a corpus. Several additional functions are implemented for the sake of language productivity analysis and for the use of richer features than just character counts.

In addition, we released a complete data set which contains analogical clusters and analogical grids built on 1,000 corresponding lines in 11 European languages extracted from the Europarl corpus v.3.

We hope that such module and data will be used by researchers in comparative linguistic studies, in re-inflection tasks or other tasks of Natural Language Processing. We hope that the tools provided in the module will be used for the study of other languages than those of the resource released.

## 5. Acknowledgements

This work was supported by a JSPS Grant, Number 15K00317 (Kakenhi C), entitled Language productivity: efficient extraction of productive analogical clusters and their evaluation using statistical machine translation.

[3] In (Chan, 2008, p. 79), saturation is the maximal proportion of word forms attested for any one lemma of a given paradigm. Here we use the term for each entire table.

## 6. Bibliographical References

Chan, E. (2008). *Structures and distributions in morphology learning*. Ph.D. thesis, University of Pennsylvania.

Fam, R. and Lepage, Y. (2016). Morphological predictability of unseen words using computational analogy. In *Proceedings of the Computational Analogy Workshop at the 24th International Conference on Case-Based Reasoning (ICCBR-CA-16)*, pages 51–60, Atlanta, Georgia.

Fam, R. and Lepage, Y. (2017). A study of the saturation of analogical grids agnostically extracted from texts. In *Proceedings of the Computational Analogy Workshop at the 25th International Conference on Case-Based Reasoning (ICCBR-CA-17)*, pages 11–20, Trondheim, Norway.

Hathout, N. (2008). Acquisition of the morphological structure of the lexicon based on lexical similarity and formal analogy. In *Proceedings of the 3rd Textgraphs workshop on Graph-based Algorithms for Natural Language Processing*, pages 1–8, Manchester, UK, August. Coling 2008 Organizing Committee.

Hathout, N. (2009). Acquisition of morphological families and derivational series from a machine readable dictionary. *CoRR*, abs/0905.1609.

Lepage, Y. (1998). Solving analogies on words: an algorithm. In *Proceedings of the 17th international conference on Computational linguistics (COLING 1998)*, volume 1, pages 728–734. Association for Computational Linguistics.

Lepage, Y. (2014). Analogies between binary images: Application to Chinese characters. In Henri Prade et al., editors, *Computational Approaches to Analogical Reasoning: Current Trends*, pages 25–57. Springer, Berlin, Heidelberg.

Mansion, J. E., (1981). *Harrap's New Shorter French and English Dictionary*. George G. Harrap Co. Ltd, London, Paris, Stuttgart.

Neuvel, S. and Fulop, S. A. (2002). Unsupervised learning of morphology without morphemes. In *Proceedings of the ACL-02 Workshop on Morphological and Phonological Learning*, pages 31–40. Association for Computational Linguistics, July.

Singh, R. and Ford, A. (2000). In praise of Sakatayana: some remarks on whole word morphology. In Rajendra Singh, editor, *The Yearbook of South Asian Languages and Linguistics-200*. Sage, Thousand Oaks.

Sylak-Glassman, J., Kirov, C., Post, M., Que, R., and Yarowsky, D., (2015a). *A Universal Feature Schema for Rich Morphological Annotation and Fine-Grained Cross-Lingual Part-of-Speech Tagging*, pages 72–93. Springer International Publishing, Cham.

Sylak-Glassman, J., Kirov, C., Yarowsky, D., and Que, R. (2015b). A language-independent feature schema for inflectional morphology. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 674–680, Beijing, China, July. Association for Computational Linguistics.

## 7. Language Resource References

Christodouloupoulos, Christos. (2015). *A massively parallel corpus: the Bible in 100 languages*. Distributed via github[4], bible-corpus v1.8.

Christo Kirov and John Sylak-Glassman and Roger Que and David Yarowsky. (2016). *UniMorph Data*. UniMorph Project, distributed via web[5].

Koehn, Philipp. (2007). *European Parliament Proceedings Parallel Corpus*. Distributed via web[6], Europarl v3.

[4]github.com/christos-c/bible-corpus

[5]unimorph.org

[6]statmt.org/europarl/