# Semantico-formal resolution of analogies between sentences

**Yves Lepage**

Waseda University
2-7 Hibikino, Wakamatsu-ku, Kitakyushu-shi, 808-0135 Fukuoka-ken, Japan
yves.lepage@waseda.jp

**Abstract**

This paper proposes a way of solving analogies between sentences by combining existing techniques to solve formal analogies between strings and semantic analogies between words. Experiments on sentences from the Tatoeba corpus are conducted and a dataset of more than five thousand semantico-formal analogies is released.

## 1. Goal of the paper

Formal analogies between strings are puzzles of the general type: *abc* : *abbccd* :: *efg* : $x$ (solution: *effggh*) (Hofstadter and the Fluid Analogies Research Group, 1994) or *król* : *królowa* :: *kr* : $x$ (solution: *krowa*). No meaning is attached to the strings in such analogies. Techniques have been proposed to solve puzzles that involve prefixing, infixing and parallel infixing (Lepage, 1998; Langlais et al., 2009), e.g., *kataba* : *kātib* :: *sakana* : $x$ (solution: *sākin*). By considering sentences as being strings of words, these techniques solve analogies like (1).

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| *You will see the man next week.* | : | *I see the woman this week.* | :: | *You will meet the man next month.* | : | $x$ | (1) |

The solution is: $x$ = *I meet the woman this month.*

Formal analogies do not care about meaning. They are different from semantic analogies, the over-repeated example of which is *man* : *woman* :: *king* : *queen* (Mikolov et al., 2013). Semantic analogies recently became popular because vector representations of words, especially word embeddings (Mikolov et al., 2013; Pennington et al., 2014; Arora et al., 2016; Bojanowski et al., 2017; Peters et al., 2018), can be used to solve them.

This paper proposes a way to combine resolution of formal analogies between strings with resolution of semantic analogies between words so as to solve (some) analogies between sentences, like the one in (2).

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| *You will see the man next week.* | : | *I saw the woman last week.* | :: | *You will meet the King tomorrow.* | : | $x$ | (2) |

The solution should be: $x$ = *I met the Queen yesterday.* Observe the change in tenses, from future to preterit, on irregular verbs, and the corresponding change in time from *next week* to *last week* expressed for days by the substitution of *tomorrow* with *yesterday*. Of course, as expected, the male / female opposition has been reflected when solving *man* : *woman* :: *King* : $x$ $\Rightarrow$ $x$ = *Queen.*

## 2. Semantic analogies

### 2.1. Vector arithmetic

Let us recall the simplest technique used to solve analogies between words in word embedding spaces. It bases on vector arithmetic. If $\overrightarrow{w}$ notes the vector corresponding to the word $w$, the resolution of the analogical equation $A : B :: C : x$, where $A$, $B$, $C$ and $x$ are words, is performed in two steps. First, a vector, $v$, is built as in (3).

$$v = \overrightarrow{B} - \overrightarrow{A} + \overrightarrow{C} \tag{3}$$

Then, the solution $x$ is defined as the word in the embedding model which maximises the cosine similarity to $v$, i.e., $\overrightarrow{x} = \arg\max_w \cos(\overrightarrow{w}, v)$, where $w$ ranges over all the words in the vector space.[1]

### 2.2. Extension to sets of words

It is possible to extend the above-mentioned use of vector arithmetic by considering sets of words. From three sets of words, $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$, one can always form the vector $v$ defined in Equation (4).

$$v = \sum_{w_B \in \mathcal{B}} \overrightarrow{w_B} - \sum_{w_A \in \mathcal{A}} \overrightarrow{w_A} + \sum_{w_C \in \mathcal{C}} \overrightarrow{w_C} \tag{4}$$

The word $x$ whose vector maximises the cosine similarity to $v$, is considered the solution of the analogical equation between the given sets of words: $\mathcal{A} : \mathcal{B} :: \mathcal{C} : x$. Note that, here, on the contrary to $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$, $x$ is a single word.

This simple extension allows us to solve small analogies like: $\{$*will*, *see*$\}$ : $\{$*saw*$\}$ :: $\{$*will*, *meet*$\}$ : $x$ and obtain *met* as the solution. Similarly, the equation $\{$*next*, *week*$\}$ : $\{$*last*, *week*$\}$ :: $\{$*tomorrow*$\}$ : $x$ has *yesterday* as its solution.

However, this extension does not answer the case where the solution of the analogical equation is longer than one word. It does not allow us to get the expected answer to analogies like $\{$*tomorrow*$\}$ : $\{$*yesterday*$\}$ :: $\{$*next*, *week*$\}$ : $x$ where the expected solution would be a sequence of two words, namely *last week*. And more importantly, it does not apply to the resolution of analogies between complete sentences.

## 3. Formal analogies

### 3.1. Traces

We now review methods to solve formal, not semantic, analogies between strings. There exist two trends for solving formal analogies between strings. The first one (Langlais and Yvon, 2008; Langlais et al., 2009) uses the

[1] The validity of this linear conception has been questioned in (Drozd et al., 2016) and other formulae have been proposed.

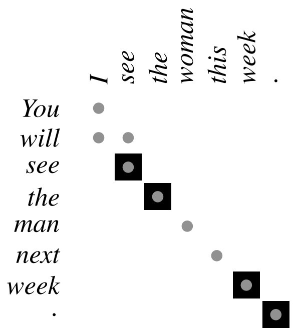

Figure 1: Traces between two sentences using LCS distance. The matrix cells are either black (equality) or white.

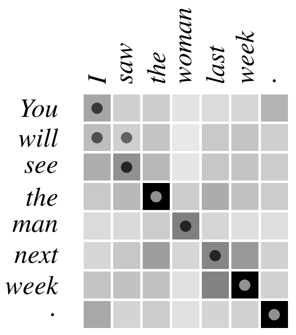

Figure 2: Traces between two sentences using word embedding distance. The shades of grey reflect similarity: the darker, the more simlar. The top sentence is different from Figure 1, but traces are similar.

notion of shuffle of strings to produce a solution while the second one (Lepage, 1998; Lepage, 2003) bases on the computation of edit distances between strings. The two trends share some abstract commonalities, but we will concentrate on the second one. There, the crucial notion is that of a *trace*, i.e., a sequence of edit operations, including copying, to apply so as to transform one string into another. The classical algorithms to compute an edit distance and a trace are found in (Wagner and Fischer, 1974). Another possible algorithm to deliver a trace is given in (Hirschberg, 1975).

An illustration of traces is given in Figure 1. The two strings are actually sentences, the words of which are just symbols compared for equality. Equality is shown by black squares. The grey points visualise the traces, i.e., the shortest paths linking the top left cell to the bottom right one in the matrix, which minimises the number of edit operations needed to transform the sentence on the left into the sentence at the top. There are two possible traces here.

### 3.2. LCS edit distance and parallel traversal of traces

An illustration of the algorithm for the resolution of formal analogies between strings proposed in (Lepage, 1998) (based on (Itkonen and Haukioja, 1997)) is given in Figure 3. Here again, the strings are sentences with words compared for equality. The arithmetic formula $b - a + c$ is applied to compute the word at hand in the solution from the words $a$, $b$ and $c$ found in $A$, $B$ and $C$ while reading the two traces, between $A$ and $B$ and between $A$ and $C$, in parallel. Note, however, that the formula makes sense only when $a = b$ or $a = c$.

Leaving copying apart (cost of 0), the edit operations used in the computation of the trace are reduced to two: insertion and deletion, each with a cost of 1. Consequently, the substitution of a symbol with another one has a cost of 2, because this corresponds to a deletion and an insertion, each of a cost of 1.

The edit distance with insertion and deletion only corresponds to the similarity between strings classically defined as their longest common subsequence (LCS) through Equation (5). For this reason, it is called the LCS distance.

$$d(A,B) = |A| + |B| - 2 \times s(A,B) \tag{5}$$

In Equation (5), $d$ is the LCS distance between two strings, $s$ is their similarity, i.e., the length of their longest common subsequence (LCS), and $|s|$ denotes the length of a string $s$. With the additional notation that $|s|_a$ denotes the number of occurrences of character $a$ in string $s$, an analogy between strings can be characterised by the system in (6).

$$A : B :: C : x \Rightarrow \begin{cases} d(x,C) = d(A,B) \\ d(x,B) = d(A,C) \\ \mid x \mid_a = \mid B \mid_a - \mid A \mid_a + \mid C \mid_a \end{cases} \tag{6}$$

## 4. Analogies between sentences

### 4.1. Distance between words and sentences

The similarity between two words $w_1$ and $w_2$ represented by their vectors in a word embedding space of a given dimensionality $n$, is classically computed as the cosine of their corresponding vectors, as in Equation (7).

$$s(w_1, w_2) = \cos(\overrightarrow{w_1}, \overrightarrow{w_2}) \tag{7}$$

The distance between two words can then be taken as the Euclidian distance between the two points pointed by the word vectors. Under the usual assumption that all word vectors are located on the unit $n$-sphere[2], their distance is a function of the cosine of the word vectors as given by Eq. (8).[3]

$$\begin{aligned} d(w_1, w_2) &= \|\overrightarrow{w_1} - \overrightarrow{w_2}\| \\ &= \sqrt{2} \times \sqrt{1 - \cos(\overrightarrow{w_1}, \overrightarrow{w_2})} \end{aligned} \tag{8}$$

[2] This is usually the case, as normalisation is applied on raw vectors, learnt from a corpus, before any use.

[3] This is proven as follows:

$$\begin{aligned} \|\overrightarrow{w_1} - \overrightarrow{w_2}\|^2 &= (\overrightarrow{w_1} - \overrightarrow{w_2})^\top (\overrightarrow{w_1} - \overrightarrow{w_2}) \\ &= \|\overrightarrow{w_1}\|^2 + \|\overrightarrow{w_2}\|^2 - 2 \times \overrightarrow{w_1}^\top \overrightarrow{w_2} \\ &= \|\overrightarrow{w_1}\|^2 + \|\overrightarrow{w_2}\|^2 - 2 \, \|\overrightarrow{w_1}\| \|\overrightarrow{w_2}\| \cos(\overrightarrow{w_1}, \overrightarrow{w_2}) \end{aligned}$$

With all norms being equal to 1, the equality is rewritten as follows, hence Eq. (8) above.

$$\|\overrightarrow{w_1} - \overrightarrow{w_2}\|^2 = 2 \times (1 - \cos(\overrightarrow{w_1}, \overrightarrow{w_2}))$$

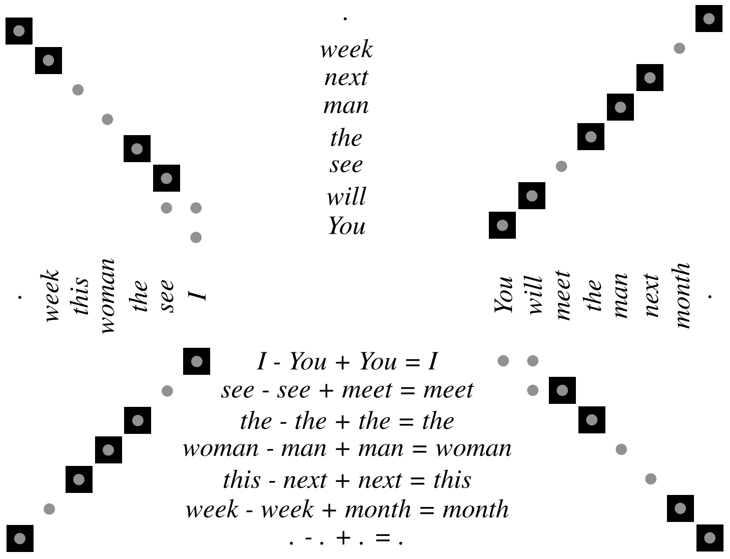

Figure 3: Resolution of a formal analogy between sentences. Observe that the top-left sentences and their traces are the same as in Figure 1, up to mirroring.

The maximal value for this distance is 2. This makes it look like the LCS distance, where the distance between two different characters is 2 (see Section 3.2.). Between words, a distance of 2 would correspond to deleting a word and inserting its exact opposite word on the unit $n$-sphere.[4]

The matrix of the distances between the words in two sentences can be visualised as in Fig. 2. Because we deal with a true mathematical distance between words, the edit distance between sequences of words is also a true mathematical distance. It can thus be computed using the standard algorithm (Wagner and Fischer, 1974). The traces can also be computed in the standard way or directly (Hirschberg, 1975). In Figure 2, the cells with a gray dot in their centre are the cells on the traces. There are two possible traces.

## 4.2. Analogy-compatible decomposition of a quadruple of strings

A decomposition of a pair of strings $(A, B)$, each into two parts $(A_1.A_2, B_1.B_2)$ such that $A = A_1.A_2$, $B = B_1.B_2$ and such that at most one of the lengths of $A_1$, $A_2$, $B_1$ and $B_2$ is null, always verifies (9).

$$d(A_1.A_2, B_1.B_2) \leq d(A_1, B_1) + d(A_2, B_2) \quad (9)$$

The equality is only reached on traces.[5] For that reason, we say that a decomposition is *trace-compatible* if it verifies Equation (10).

$$d(A_1.A_2, B_1.B_2) = d(A_1, B_1) + d(A_2, B_2) \quad (10)$$

[4] However, in practice, word vectors are not evenly distributed on the unit $n$-sphere. For instance, in the English vector space pre-trained using FastText (see Section 5.2.), the word closest to the opposite point of $\overrightarrow{Queen}$ is $\overrightarrow{component.}$ (with a glued full stop). But we find: $\cos(\overrightarrow{Queen}, \overrightarrow{component.}) = -0.203$, which is far from $-1.0$.

[5] Combined with the mirror of strings, this property is used in (Hirschberg, 1975) to directly find an optimal alignment between two strings.

An *analogy-compatible* decomposition of a quadruple of strings $(A, B, C, x)$ is a quadruple of decompositions $(A_1.A_2, B_1.B_2, C_1.C_2, x_1.x_2)$ where each individual decomposition $(A_1.A_2, B_1.B_2)$, $(A_1.A_2, C_1.C_2)$, $(B_1.B_2, x_1.x_2)$ and $(C_1.C_2, x_1.x_2)$ is trace-compatible. Given a quadruple of strings $(A, B, C, x)$, we note $\tau(A, B, C, x)$ the set of all analogy-compatible decompositions.

## 4.3. Semantico-formal resolution of analogies between sentences

To solve an analogy $A : B :: C : x$ between the sentences $A$, $B$ and $C$, we first compute the traces between $A$ and $B$ and between $A$ and $C$, using the distance between words introduced in Section 4.1.. We then explore the traces between $A$ and $B$ and between $A$ and $C$, in parallel, in a way which is similar to the resolution of formal analogies (Section 3.2.), so as to build the set of analogy-compatible decompositions, introduced above (Section 4.2.). There are two cases:

- Either $\tau(A, B, C, x)$ is empty; if the length of $x$ is greater than 1, we are unable to solve the analogy; but if the length of $x$ is 1, i.e., if $x$ is a single word, the method presented in Section 2.2. can be applied to solve the analogy between the sets of words in $A$, $B$ and $C$.
- Or $\tau(A, B, C, x)$ is not empty and for any decomposition $(A_1.A_2, B_1.B_2, C_1.C_2, x_1.x_2)$, we can try and solve $A_1 : B_1 :: C_1 : x_1$ and $A_2 : B_2 :: C_2 : x_2$. The recursion will ultimately end up on the first case.

The result of the semantico-formal resolution of the example analogy between sentences given in Section 1. using the procedure described above is illustrated in Figure 4. It succeeds in delivering the expected solution: *I met the Queen yesterday.*

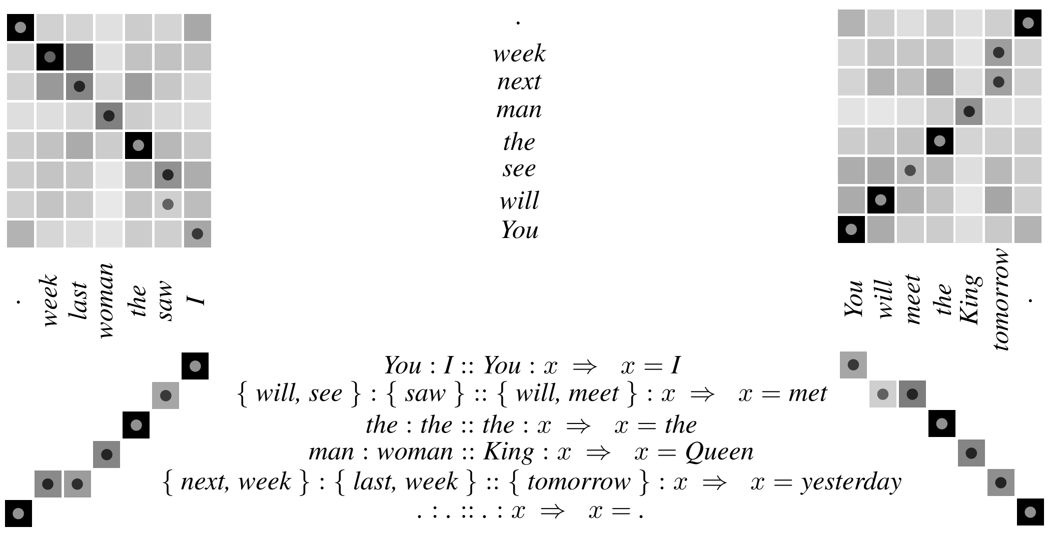
.
week
next
man
the
see
will
You
.
week
last
woman
the
saw
I
You
will
meet
the
King
tomorrow
.
You : I :: You : x ⇒ x = I
{ will, see } : { saw } :: { will, meet } : x ⇒ x = met
the : the :: the : x ⇒ x = the
man : woman :: King : x ⇒ x = Queen
{ next, week } : { last, week } :: { tomorrow } : x ⇒ x = yesterday
. : . :: . : x ⇒ x = .

Figure 4: Semantico-formal resolution of an analogy between sentences. Observe that the top-left matrix is the same as in Figure 2, up to mirroring.

## 5. Experiments and released dataset

We perform experiments on English data to confirm the fact that our technique is indeed able to solve analogies between sentences. Basically, we try to solve all analogies between all triples of sentences extracted from a resource rich in similar sentences. We of course apply restrictions.

Firstly, so as to reduce the number of possible triples, we only consider sentences shorter than 10 words. Secondly, so as to avoid the problems mentioned at the end of Section 2.2., we impose that, for an analogy $A : B :: C : x$, the length of $A$ be equal to or greater than the length of $B$, and the same for $B$ and $C$. Thirdly, we impose that the number of words in common between $B$ and $A$ be higher than two thirds of the length of $A$, and the same between $C$ and $A$.

### 5.1. Difficulties of assessment

A difficulty in assessing the technique proposed in this paper lies in the fact that vector arithmetic followed by the determination of the closest word always delivers a solution. Such solutions simply make no sense in the immense majority of the cases. As an example, with our data, the analogy *suit* : *pool* :: *sister* : $x$ yields the highly questionable solution $x$ = *aunt*. This problem is inherited for sentences by the technique proposed in this paper. It delivers a solution for a very large number of analogies between sentences, without a guarantee in meaning. For instance, in our experiments, *I do not have a suit .* : *I do not have medical training .* :: *I have not got a chance .* : $x$ yields the solution $x$ = *I have not got medical opportunity .*

Another difficulty is the fact that the sequence of words produced may result in ungrammatical sentences (e.g. *I do not have go answers questions .*). This is the well-known problem of *over-generation*. A similar problem is that the solution sentences may contain words which actually belong to the embedding space but are misspelled words or OCR errors from the texts the word embedding space was trained from. As a example, in our experiments, we got: $x$ = *I do not recommend Engish !* (note the absence of *l*) as the solution of an analogy.

Assessing the validity of the obtained analogies would thus require a heavy and tedious work by human judges. Because of the highly subjective assessment required, we do not except any reasonable inter-judge agreement. For that reason, we propose to restrict ourselves to those analogies which deliver a sentence already present in the resource. In this way, any solution of an analogy should be a valid sentence.

### 5.2. Used datasets

We use the English sentences from the English-French Tatoeba corpus.[6] There are 92,062 sentences shorter than 10 words with an average length of 6.9 word (std.dev. of 1.7 word). The total number of different words is 13,813.

This corpus is well-fitted for our work as it exhibits al large number of similar sentences with simple commutations like masculine / feminine, affirmative / negative, etc., as illustrated in the following real example: *I do not know his address .* : *I do not know her address .* :: *I know his address .* : *I know her address .*

As for word embeddings, we use the English vectors trained with FastText and released at LREC 2018, among other languages (Bojanowski et al., 2017; Grave et al., 2018).[7] Out of the 2 million words, we retain 1,192,424 words by filtering out long numbers, ill-formed words, etc.

### 5.3. Released dataset

In total with all the restrictions described above, we obtained 5,607 semantico-formal analogies. Some of them are given in Table 1. These semantico-formal analogies are released as a public resource.[8].

[6] `https://tatoeba.org/` and `http://www.manythings.org/anki/`

[7] `https://fasttext.cc/docs/en/pretrained-vectors.html` Word embeddings for 294 languages.

[8] `http://lepage-lab.ips.waseda.ac.jp/` Projects > kakenhi-18K11447 > “Experimental Results”

|  |  |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|
| *There's hardly any coffee left in the pot.* | : | *There's almost no coffee left in the pot.* | :: | *There's hardly any water in the bucket.* | : $x$ | $\Rightarrow$ | $x =$ | *There's almost no water in the bucket.* |
| *I do not know what to say about that .* | : | *I do not know what to do now .* | :: | *I do not know about that .* | : $x$ | $\Rightarrow$ | $x =$ | *I do not think so .* |
| *You 're not from around here , are you ?* | : | *You 're not staying here , are you ?* | :: | *You 're confused again , are n't you ?* | : $x$ | $\Rightarrow$ | $x =$ | *You 're disappointed , are n't you ?* |
| *There is an urgent need for a new system .* | : | *There is an urgent need for blood donations .* | :: | *There is an urgent need for experienced pilots .* | : $x$ | $\Rightarrow$ | $x =$ | *There is an urgent need for volunteers .* |
| *I do not know his name .* | : | *I do not know her address .* | :: | *I ca n't remember his name .* | : $x$ | $\Rightarrow$ | $x =$ | *I ca n't remember her address .* |
| *It 's really not that interesting .* | : | *It 's really not that hot .* | :: | *It 's not that bad .* | : $x$ | $\Rightarrow$ | $x =$ | *It 's not that cold .* |

Table 1: Examples of semantico-formal analogies from the released dataset

## 6. Conclusion

In this paper, we proposed an approach to solve analogies between sentences which is different from those which try to make direct use of neural networks (Zhao and Lepage, 2018), or try to directly learn sentence representations (Pagliardini et al., 2018). Our approach combines semantic word analogies with formal string analogies.

There are still some problems left. For instance, how to solve analogies like *last week* : *in two weeks* :: *yesterday* : *the day after tomorrow* where the lengths in words do not verify the analogical arithmetic relation, i.e., $2 - 3 \neq 1 - 4$?

We discovered a pressing need for cleaning word embedding models from spurious words. In contrast to that, for languages with a much richer morphology than English, another problem will arise from the word embedding resources used: what if a declined or conjugated word form is missing from the embedding space?

The difficulties in assessing the dataset arise from the fact that semantic analogies between words are in fact still highly unreliable. The reliability of analogies between sentences heavily depends on that. Datasets like the ones released in (Mikolov et al., 2013) (Google set) or in (Drozd et al., 2016) (BATS v3.0) have indeed an insufficiently small coverage relatively to the very large possibilities opened by our technique.

## 7. Acknowledgement

Work supported by JSPS Grant 18K11447 (Kakenhi C) “Self-explainable and fast-to-train example-based machine translation using neural networks”.

## 8. References

Arora, S., Y. Li, Y. Liang, T. Ma, and A. Risteski, 2016. A latent variable model approach to PMI-based word embeddings. *TACL*, 4:385–399.

Bojanowski, P., E. Grave, A. Joulin, and T. Mikolov, 2017. Enriching word vectors with subword information. *TACL*, 5:135–146.

Drozd, A., A. Gladkova, and S. Matsuoka, 2016. Word embeddings, analogies, and machine learning: Beyond king - man + woman = queen. In *COLING 2016*.

Grave, E., P. Bojanowski, P. Gupta, A. Joulin, and T. Mikolov, 2018. Learning word vectors for 157 languages. In *LREC 2018*.

Hirschberg, D. S., 1975. A linear space algorithm for computing maximal common subsequences. *Comm. of the ACM*, 18(6):341–343.

Hofstadter, D. and the Fluid Analogies Research Group, 1994. *Fluid Concepts and Creative Analogies*. New York: Basic Books.

Itkonen, E. and J. Haukioja, 1997. A rehabilitation of analogy in syntax (and elsewhere). In *Metalinguistik im Wandel: die kognitive Wende in Wissenschaftstheorie und Linguistik*. Peter Lang, pages 131–177.

Langlais, P. and F. Yvon, 2008. Scaling up analogical learning. In *Coling 2008*.

Langlais, P., P. Zweigenbaum, and F. Yvon, 2009. Improvements in analogical learning: application to translating multi-terms of the medical domain. In *EACL 2009*.

Lepage, Y., 1998. Solving analogies on words: an algorithm. In *COLING-ACL-98*, volume I.

Lepage, Y., 2003. *De l'analogie rendant compte de la commutation en linguistique*. Habilitation thesis, Université de Grenoble.

Mikolov, T., W.-T. Yih, and G. Zweig, 2013. Linguistic regularities in continuous space word representations. In *NAACL-HLT 2013*.

Pagliardini, M., P. Gupta, and M. Jaggi, 2018. Unsupervised learning of sentence embeddings using compositional n-gram features. In *NAACL-HLT 2018*.

Pennington, J., R. Socher, and C. D. Manning, 2014. GloVe: Global vectors for word representation. In *EMNLP 2014*.

Peters, M. E., M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, 2018. Deep contextualized word representations. In *NAACL-HLT 2018*.

Wagner, R. A. and M. J. Fischer, 1974. The string-to-string correction problem. *J. of the ACM*, 21(1):168–173.

Zhao, T. and Y. Lepage, 2018. Context encoder for analogies on strings. In *PACLIC 32*.