Solving Analogical Equations Using Probabilities

Gyeonghun Kim, Omi Keisuke and Yves Lepage
The Graduate School of Information, Production and Systems
Waseda University
kei2407@ruri.waseda.jp, kei.imp@ruri.waseda.jp, yves.lepage@waseda.jp

1 Introduction

1.1 Motivation

In [2] an algorithm to solve analogical equations using three constraints is proposed. The results show a recall of 100% but only a precision of 0.14%. This means that many spurious solutions are included in the output. In this paper, we use probabilities to improve the precision of the solutions. In addition, we use edit distances to further improve precision.

1.2 Analogical Equation

Proportional analogies are a general relationship between four objects, A, B, C, and D. An analogy $A:B::C:D$ states that ”A is to B as C is to D”. For example, 食べる : 食べます :: 認める : 認めます [2]. In modern linguistics, analogy is considered to be a synchronic operation by which given two forms of a given word, and only one form of a second word, the fourth missing form is coined. For example, 食べる : 食べます :: 認める : x $\Rightarrow$ $x =$ 認めます.
Solving analogical equations as the above one, between strings of symbols, is an important core method in example-based machine translation (EBMT) following the proposal by Nagao in 1984 [6]. In 2005, Lepage and Denoual [5] refined the proposal. In their framework, solving analogical equations is crucial. Let us suppose that (A,A′), (B,B′), (C, C′) are in the translation table of an EBMT system. To translate D, if C can be obtained as the solution of the analogical equation between A, B, and D in the source side, then, D′ can be output from A′, B′, and C′ in the target side, with the result that D′ is the translation of D.

$$\begin{tikzcd}
\mathrm{A} \arrow[dd, leftrightarrow] & : & \mathrm{B} \arrow[dd, leftrightarrow] & :: & \mathrm{x} \arrow[d] & : & \mathrm{D} \arrow[dd, leftrightarrow] \\
 & & & & C \arrow[d, leftrightarrow] & & \\
\mathrm{A}' & : & \mathrm{B}' & :: & \mathrm{C}' & : & \mathrm{y} \arrow[d] \\
 & & & & & & \mathrm{D}'
\end{tikzcd}$$

2 Previous Work

Analogy has been mainly used in linguistics during the 19th and 20th centuries. Lepage [3] provides an algorithm for the resolution of analogical equations to solve proportional analogies between strings of symbols. The algorithm is based on an edit distance with constraints. Precisely, it relies on the following implication:

$$A:B::C:D \Rightarrow \begin{cases} |A|_\alpha + |D|_\alpha = |C|_\alpha + |B|_\alpha, \forall\alpha \\ d(A,B) = d(C,D) \\ d(A,C) = d(B,D) \end{cases} \tag{1}$$

Ito et al. [2] refined the first line in (1) using three constraints. They did not use the second and third line with the distance constraint. The three constraints are length of the solution, number of occurrences of each symbol in the solution, and positions of symbols in the solution. We explain these three constraints below.

2.1 Length of the Solution

The length of the solution D is determined by the length of the three given terms A, B, and C. The length of the solution is given by:

$$\begin{gathered} |A| + |D| = |B| + |C| \\ \Rightarrow |D| = |B| + |C| - |A| \end{gathered} \tag{2}$$

2.2 Number of Occurrences of Symbols

The number of occurrences of symbols for the solution D is determined by their number of occurrences in the three given terms A, B, and C. The definition of the number of occurrences of the symbol a in the solution D is given by the first equation in the right part of (1):

$$\begin{gathered} |A|_a + |D|_a = |B|_a + |C|_a \\ \Rightarrow |D|_a = |B|_a + |C|_a - |A|_a \end{gathered} \tag{3}$$

2.3 Position of Symbols in the Solution

The position of the symbols in the solution is determined by their position in the three given terms A, B and C. We determine match points between A and B and between A and C in a searching area. The searching area is the diagonal of the rectangle between two terms. The width of the searching area is:

$$Area(A,B) = ||A| - |B|| + 1 \tag{4}$$

For the analogical equation $du : duzhe :: xue : D$, the width of the searching area between du and $duzhe$ is $Area(du, duzhe) = |2-5|+1 = 4$ in this case.

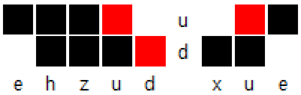

Figure 1: Searching area between terms for the analogical equation $du : duzhe :: xue : D$.

In Figure 1, the black cells show the searching areas between the terms. The red cells show the matched characters between terms in the searching areas (e.g., u and d in this case). However, in some cases, some match points do not fall in the searching area. Let us consider the analogical equation $dues : indu :: neés : D$.

The width of the searching area between *dues* and *indu* is $Area(dues, indu) = |4-4|+1 = 1$. Figure 2 shows the searching abetween two words, reas for the analogical equation *dues* : *indu* :: *neés* : *D*.

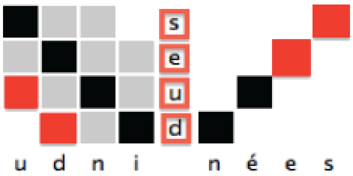

Figure 2: Searching area between terms for the analogical equation *dues* : *indu* :: *neés* : *D*.

In this case, we need to extend the searching area. When doing so, we extend the areas above and below the original area simultaneously. In Figure 2, the gray cells represent the extended area to include all match points. We call this area the extra band. In the example above, the extended area represents four squares, so the extra band size is two, and two squares are taken above and below the original searching area. With this modification, the definition of the searching area is:

$$Area(A,B) = ||A|-|B||+1+|extra\_band| \qquad (5)$$

From experiments conducted in previous research [2] the searching area necessary to find a solution is determined as:

$$\begin{array}{c} Area(A,B) = Area(C,D) \\ Area(A,C) = Area(B,D) \end{array} \qquad (6)$$

To summarize all the constraints given above, the analogical equation *du* : *duzhe* :: *xue* : *D*, the length of the solution *D* is 6, the number of occurrences of each symbol in the solution *D* is $(x:1, u:1, z:1, h:1, e:2)$ and Figure 3 shows the possibilities of placing symbols in the solution *D*. In the solution, the character *d* does not appear. Hence, we removed the possibility of having match points below character *d* from the diagram.

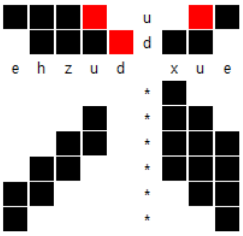

Figure 3: Searching area for the solution for the analogical equation *du* : *duzhe* :: *xue* : *D*.

3 Proposed Method

Previous research reported in [2] yielded a recall of 100%, but only produce a precision of 0.14% when applying the previous three constraints. Based on the inspection of Figure 3 and many other cases, we propose four additional constraints to eliminate spurious results. Our goal cannot be to achieve a precision of 100% on a given set of linguistic examples. For example, in *du* : *duzhe* :: *xue* : *D*, *D* might be a *xuezhe* or *xuzhee* in human thought; however, Chinese data will only propose *xuezhe*. Therefore, achieving 100% precision is clearly not a requirement, but any method improving precision is highly desirable.

3.1 Scoring by probability of characters given position

With these new constraints, we determine the character for each position in the solution by calculating the probability of a character given its position $P(Char = x\,|$pos=y). This is given by:

$$P(Char=x\,|\,pos=y) = \frac{P(Char=x)\cap P(pos=y)}{P(pos=y)} \qquad (7)$$

For the analogical equation *du* : *duzhe* :: *xue* : *D*, the possibilities have been presented in Figure 3. The probabilities for each character given its position is calculated in Table 1.

Position in D	x	u	e	z	h
D[1]	$\frac{1}{1}$				
D[2]	$\frac{1}{3}$	$\frac{2}{3}$			
D[3]	$\frac{1}{5}$	$\frac{2}{5}$	$\frac{1}{5}$	$\frac{1}{5}$	
D[4]	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{1}{5}$
D[5]		$\frac{1}{4}$	$\frac{2}{4}$		$\frac{1}{4}$
D[6]			$\frac{2}{2}$		

Table 1: Probability of a character given its position for the analogical equation *du* : *duzhe* :: *xue* : *D*.

This yields solutions with a probabilistic score. The best score is 1.00. In this method, the closer the score to 1.00, the better the solution. The score of the solution is given by:

$$Score = \Pi P(character|position) \qquad (8)$$

3.2 Scoring by probability of column given position

We determine the match points for each position in the solution. To determine a match point, we separate the characters by columns and positions. For each possibility, we calculate the probability of a column given its position $P(Col = x\,|$pos=y) by:

$$P(col=x\,|\,pos=y) = \frac{P(col=x)\cap P(pos=y)}{P(pos=y)} \qquad (9)$$

For the analogical equation *du* : *duzhe* :: *xue* : *D*, the possibilities are presented in Figure 3. The probability of a column given its position is calculated a show in Table 2. As with the prievous constraint, this method yields solutions with a probabilistic score. The initial score is 1.00. The score of a solution is given by:

$$Score = \Pi P(column|position) \qquad (10)$$

With this method, all results have the same score because each match point appears only once. This method delivers the same results as with the previous research reported in [2].

3.3 Scoring by contiguity of characters

For this method, we determine contiguous match points using the probability of column given position discussed

Position in D	B[5]	B[4]	B[3]	B[2]	B[1]	C[1]	C[2]	C[3]
D[1]						$\frac{1}{1}$		
D[2]				$\frac{1}{3}$		$\frac{1}{3}$	$\frac{1}{3}$	
D[3]			$\frac{1}{5}$	$\frac{1}{5}$		$\frac{1}{5}$	$\frac{1}{5}$	$\frac{1}{5}$
D[4]		$\frac{1}{5}$	$\frac{1}{5}$			$\frac{1}{5}$	$\frac{1}{5}$	$\frac{1}{5}$
D[5]	$\frac{1}{4}$	$\frac{1}{4}$					$\frac{1}{4}$	$\frac{1}{4}$
D[6]	$\frac{1}{2}$							$\frac{1}{2}$

Table 2: Probability of a column given its position for the analogical equation *du* : *duzhe* :: *xue* : *D*.

above. The contiguity of the results is a sequence of adjacent characters from the same term. In this way, most of contiguous match points tend to be prefixes or suffixes. For the analogical equation *du* : *duzhe* :: *xue* : *D,* Table 3 shows the results with this method. We score substrings by calculating the score of a solutions according to:

$$\begin{array}{c} Score(substring) = |Continuous\ \ seq| - 1 \\ Score = \Sigma(Score(substring)) \end{array} \tag{11}$$

Here the higher the score, the better the solution. The results are shown in Table 3. The result in boldface is the best result obtained using this method (e.g.,*xuezhe*).

Match point	Solution	score
C[1],C[2],C[3],B[3],B[4],B[5]	**xuezhe**	2 + 2 = 4
C[1],B[2],C[3],B[3],B[4],B[5]	xuezhe	0 + 0 + 0 +2 = 2
C[1],C[2],B[3],B[4],B[5],C[3]	xuzhee	1 + 2 + 0 = 3
C[1],B[2],B[3],B[4],B[5],C[3]	xuzhee	0 + 3 + 0 = 3
C[1],C[2],B[3],B[4],C[3],B[5]	xuzhee	1 + 1 + 0 + 0
C[1],B[2],B[3],B[4],C[3],B[5]	xuzhee	0 + 2 + 0 + 0 = 2
C[1],C[2],B[3],C[3],B[4],B[5]	xuzehe	1 + 0 + 0 + 1 = 2
C[1],B[2],B[3],C[3],B[4],B[5]	xuzehe	0 + 1 + 0 + 1 = 2

Table 3: The results of the Contiguity method for the analogical equation *du* : *duzhe* :: *xue* : *D*.

3.4 Scoring entropy based on reordering

In this method, we use the probability of characters given positions to calculate the entropy at each position. We choose a position according to the least entropy to determine which character it should contain. After selecting the position and the character, the porobabilities are updated. As a result, the entropies at each position are changed. This creates a new state for which the total entropy is:

$$\text{Entropy of a state} = \Sigma(\text{Entropy of position}) \tag{12}$$

Using the entropy of each state, we can trace the states used to build a solution, one position after another, as shown in Table 4, 5. Table 5 shows the state after choosing the character D[1] = C[1] = *x* and before choosing the character D[6] = C[3] = *e*. The character *x* disappears from position D[2] as it should appear only once in the solution. We thus remove this character by the constraint on the number of occurrences of symbols in the solution. Then, we recompute the probability of all characters given its position, and consequently, the entropies for each position in the solution. The equation for the entropy score of a solutions is given by:

$$Score = \Sigma(\text{Entropy of position}) \tag{13}$$

In this method, the lower the score, the better the solution.

Position in D	x	u	e	z	h	Entropy
D[1]	$\frac{1}{1}$					0.00
D[2]	$\frac{1}{3}$	$\frac{2}{3}$				0.92
D[3]	$\frac{1}{5}$	$\frac{2}{5}$	$\frac{1}{5}$	$\frac{1}{5}$		1.92
D[4]	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{1}{5}$	2.32
D[5]		$\frac{1}{4}$	$\frac{2}{4}$		$\frac{1}{4}$	1.92
D[6]			$\frac{2}{2}$			0.00
Entropy of state						6.66
Result						[, , , , ,]

Table 4: Entropy of an initial state for the analogical equation *du* : *duzhe* :: *xue* : *D*.

Position in D	x	u	e	z	h	Entropy
D[1] = '*x*'						-
D[2]		$\frac{2}{2}$				0.00
D[3]		$\frac{2}{3}$		$\frac{1}{3}$		0.92
D[4]		$\frac{1}{3}$		$\frac{1}{3}$	$\frac{1}{3}$	1.58
D[5]		$\frac{1}{3}$	$\frac{1}{3}$		$\frac{1}{3}$	1.58
D[6] = '*e*'						-
Entropy of state						4.08
Result						[x, , , , ,e]

Table 5: Entropy of an second state for the analogical equation *du* : *duzhe* :: *xue* : *D*.

4 Experiments

4.1 Data

We use linguistic examples that are true proportional analogies in their language for our test experiments. The data cover 12 languages: Japanese and Chinese and 10 other languages from the Europarl Corpus (such as German, French, or English). We use 99 basic proportional analogies in these 12 languages. Using the fundamental properties of proportional analogies, it is possible to generate seven different equivalent forms that have the same meaning. In addition, by reading from right to left, i.e., taking the mirror of strings, seven other equivalent forms can be generated. The forms of the proportional analogies [4] are:

$$\begin{array}{ll} A:B::C:D & ,\ A^{-1}:B^{-1}::C^{-1}:D^{-1} \\ A:C::B:D & ,\ A^{-1}:C^{-1}::B^{-1}:D^{-1} \\ B:A::D:C & ,\ B^{-1}:A^{-1}::D^{-1}:C^{-1} \\ B:D::A:C & ,\ B^{-1}:D^{-1}::A^{-1}:C^{-1} \\ C:A::D:B & ,\ C^{-1}:A^{-1}::D^{-1}:B^{-1} \\ C:D::A:B & ,\ C^{-1}:D^{-1}::A^{-1}:B^{-1} \\ D:B::C:A & ,\ D^{-1}:B^{-1}::C^{-1}:A^{-1} \\ D:C::B:A & ,\ D^{-1}:C^{-1}::B^{-1}:A^{-1} \end{array}$$

In the above, A^{-1} is the mirror of A (*cba* for *abc*). We generate $1584 = 99 \times 16$ linguistic examples from 99 proportional analogies. Because our algorithms are neutral relatively to the exchange of the means ($A:B::C:D \Leftrightarrow A:C::B:D$), we generate only half of the possible analogical equations. As a result, from 99 equations, we collected $1584/2 = 792$ new analogical equations. These examples constitute the same data set as previously used in [2]. Linguistic negative examples are also used. They are created based on linguistic examples. The general form of a linguistic negative is:

$$A:B \neq D:C$$

When $A:B::C:D$ is a valid analogy, $A:B::D:C$ is generally not an analogy [1]. We checked all our negative examples by hand for not being valid analogies. In this way, we generated $792 = 99 \times 8$ linguistic negative examples from 99 proportional analogies.

	Precision	Recall	F-measure	Accuracy
Character given position	4.8%	73.4%	9.0%	10.0%
same + edit distance	41.6%	87.0%	56.3%	58.1%
Column given position	0.1%	**100.0%**	0.1%	0.1%
same + edit distance	11.9%	**100.0%**	21.3%	21.2%
Contiguity	60.2%	85.5%	70.7%	54.3%
same + edit distance	**81.2%**	94.6%	**87.6%**	**87.9%**
Entropy	42.1%	68.8%	52.3%	48.4%
same + edit distance	59.8%	79.2%	68.2%	70.8%
C program[3]	33.9%	44.4%	38.4%	44.3%

Table 6: Performance of methods (Boldface of results are the best).

4.2 Applying edit distances

We also apply edit distances to filter the results of the four methods. Spurious solutions are of two types: 1. They may be theoretically valid solutions but may not exist in the language considered; 2. They may be theoretically invalid; these are truly spurious solutions.

Results	Score
yxeuuan	0.0042
xyeuuan	0.0042
xueyuan	0.0021

(a) Before applying edit distances

Results	Score
xueyuan	0.0021

(b) After applying edit distances

Table 7: Applying edit distances for the analogical equation *yi* : *xue* :: *yiyuan* : *D*.

Table 7a shows the results on the set of solutions of the analogical equation *yi* : *xue* :: *yiyuan* : *D* for the method probability of character given position. After applying the edit distance constraints, the original set of results is redefined as a new set where only the best results are included as shown in Table 7b. The edit distance constraint removes spurious solutions from the results (e.g., the top two in Table 7a). After these are removed, we select the best solution, i.e., the one with. For the example above, only one solution remains after applying edit distance. It thus becomes the best solution by default. The application of edit distances generally improves the precision of our methods.

4.3 Results

Although our goal is mainly to improve the precision compared to previous research [2], we measure our results not only using precision and recall but also using F-measure and accuracy. We also measure the results against those of a previous C program [3] as a baseline. This C program had the drawback of outputting only the first solution it finds. All the results are summarized in Table 6.

As said already, the method in [2] yields a recall of 100%, but a very low precision of 0.14%. For recall, our Column given position method with edit distance produces the best results. The other three methods, Probability of character given position, Entropy and Contiguity, improved precision, F-measure and accuracy. Applying edit distances always improves the precision over the basic method. Applying the edit distance with character given position, contiguity and entorpy yields better results than the C program [3].

5 Conclusions

Our proposed methods improved results in solving analogical equations between strings of symbols. We obtained higher precision, but lower recall. Contiguity with edit distance yields the best results in precision, F-measure and accuracy that we could obtain until now.

References

[1] Meriam Bayoudh, Henri Prade, and Gilles Richard. Evaluation of analogical proportions through Kolmogorov complexity. *Knowledge-Based System*, 29: 20–30, 2012.

[2] Maiko Ito, Gyeonghun Kim, and Yves Lepage. A study of algorithms to solve analogical equations between strings of symbols (in Japanese). In *Proceedings of the 18th Japanese National Conference in Natural Language Processing*, pages 296–299, Nagoya, March 2013.

[3] Yves Lepage. Solving analogies on words: an algorithm. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics (COLING-ACL 98), Volume 1*, pages 735–735, Montreal, Quebec, Canada, August 1998.

[4] Yves Lepage. Analogy and formal languages. *Electronic Notes in Theoretical Computer Science*, 53: 180–191, 2004.

[5] Yves Lepage and Etienne Denoual. Purest ever example-based machine translation: Detailed presentation and assessment. *Machine Translation*, 19(3-4): 251–282, 2005.

[6] Makoto Nagao. A framework of a mechanical translation between Japanese and English by analogy principle. In *Proceedings of the International NATO Symposium on Artificial and Human Intelligence*, pages 173–180, New York, NY, USA, 1984. Elsevier North-Holland, Inc. ISBN 0-444-86545-4.