Classifying and completing word analogies by machine learning

Suryani Lim[1] Henri Prade[2] Gilles Richard[2]

1. Federation University, Churchill, Australia

2. IRIT, CNRS & Université Paul Sabatier, Toulouse, France

Abstract

Analogical proportions are statements of the form 'a is to b as c is to d', formally denoted $a : b :: c : d$. They are the basis of analogical reasoning which is often considered as an essential ingredient of human intelligence. For this reason, recognizing analogies in natural language has long been a research focus within the Natural Language Processing (NLP) community. With the emergence of word embedding models, a lot of progress has been made in NLP, essentially assuming that a word analogy like $man : king :: woman : queen$ is an instance of a parallelogram within the underlying vector space. In this paper, we depart from this assumption to adopt a machine learning approach, i.e., learning a substitute of the parallelogram model. To achieve our goal, we first review the formal modeling of analogical proportions, highlighting the properties which are useful from a machine learning perspective. For instance, the postulates supposed to govern such proportions entail that when $a : b :: c : d$ holds, then seven permutations of a, b, c, d still constitute valid analogies. From a machine learning perspective, this provides guidelines to build training sets of positive and negative examples. Taking into account these properties for augmenting the set of positive and negative examples, we first implement word analogy classifiers using various machine learning techniques, then we approximate by regression an analogy completion function, i.e., a way to compute the missing word when we have the three other ones. Using a GloVe embedding, classifiers show very high accuracy when recognizing analogies, improving state of the art on word analogy classification. Also, the regression processes usually leads to much more successful analogy comple-

Email address: `suryani.lim@federation.edu.au` (Suryani Lim[1] Henri Prade[2] Gilles Richard[2])

tion than the ones derived from the parallelogram assumption.

Keywords: Analogy, analogical proportions, natural language, word embedding

1. Introduction

It is widely agreed that human intelligence partly relies on the ability to think about relations between things, rather than simply about individual entities. Analogical reasoning is one of the facets of this ability [22]. A well-known instance of this kind of reasoning is based on analogical proportions, which are statements of the form 'a is to b as c is to d', usually denoted $a : b :: c : d$. For instance, following the historical example of Rumelhart and Abrahamson [51] "man is to king as woman is to queen" or another well-known example, "electrons are to the nucleus as planets are to the sun" are standard analogical proportions. It can then be considered that $a : b :: c : d$ holds if there is a match between two high-level relations [17]: the link between a and b matches in some sense the link between c and d. For instance, electrons are attracted by the nucleus as planets are attracted by the sun, leading to $electrons : nucleus :: planets : sun$ where the underlying relation is "orbit around". That is why most computational models of analogical reasoning assume that a binary relation exists between the pairwise components of an analogical proportion. Unfortunately, there is no clear knowledge about how these abstract relations are established by a human brain [34]. Very often, representations of formal relations are given in a top-down manner by assuming a grammar of relations (see [33] for instance). Or these binary relations are extracted from data as it is the case in [42] for instance. But very few of these models of relational learning have been applied to real-life datasets such as the ones which are currently manipulated by the NLP community.

With the emergence of word embedding models [37, 43], a lot of progress has been made. However, these models provide no hints on how to extract (binary) relations between words; for instance, we could represent the words "man", "king", "woman" and "queen" using an embedding model, but being able to represent these words in a model is insufficient to extract the binary relation "gender" between these words. Basic relational reasoning and basic inference can be achieved without explicit representations of relations. Let us clarify this assertion by focusing on the case of word analogies. In the embedding context, a word w is represented as a real-valued vector $embed(w)$ in a low[1]-dimensional vector space

[1]Low with regards to the common dimensions used in modern machine learning.

$\mathbb{R}^n$ (typically $n \in \{50, 100, 200, 300\}$). A corpus W of words is then embedded as a discrete subset $embed(W)$ of $\mathbb{R}^n$. These embedding techniques are supposed to ensure two essential properties:

p1) Vectors representation for words having a similar semantic are located close together in the final real-valued vector space, e.g., $embed(man)$ should be close to $embed(boy)$.

p2) Linguistic relations, for example, are assumed to be consistently encoded as a particular difference vector in the underlying vector space, e.g.,

$$embed(man) - embed(king) \approx embed(woman) - embed(queen)$$

If these assumptions are valid, it becomes an easy game to solve word analogies: the word x solution of the analogical equation $man : king :: woman : x$ should be such that:

$$embed(x) = embed(woman) - (embed(man) - embed(king))$$

This comes to consider $embed(man), embed(king), embed(woman), embed(x)$ as the summits of a parallelogram in $\mathbb{R}^n$. Since the $\mathbb{R}^n$'s are continuous spaces, the operation $embed(woman) - (embed(man) - embed(king))$ is unlikely to land exactly in $embed(W)$, i.e.,

$$embed(woman) - (embed(man) - embed(king)) \notin embed(W)$$

The solution is then to look for the most similar element (in terms of cosine similarity) lying in $embed(W)$. Back to the previous example, these methods lead to $queen$ as the most similar word embedding of

$$embed(woman) - (embed(man) - embed(king))$$

Despite having their own drawbacks [30], these models still achieved much success in solving word analogies. In fact, the methods based on such techniques consistently outperform traditional methods by a significant margin. In this paper, we want to build upon this success but without making any assumption that the two previous properties p1 and p2 are valid. Therefore, to deal with word analogies, we depart from the parallelogram approach. We do not assume anything about the vector interpretation of word analogies. Then, our aim in this paper is to investigate if we could learn a modeling of analogical proportion that is better than the parallelogram view which dates back to [51]. This is why we do not deal with the learning of the embedding, even if it is also a relevant machine learning task as it would be the case in a computational linguistic perspective (as in [4]). In fact, the limitations of the parallelogram view may be either due to the imperfection of the embedding, or due to the fact that the parallelogram would not be a perfect model of analogical proportion, or both.

Here, we use an existing embedding, GloVe [44], and we focus on the search for a suitable model for analogical proportions between words. In fact, GloVe is not unique; there is at least one recent well-known embedding model developed in [36]. Nevertheless, it appears that the performance of the other related works is more related to the formula defining the analogical proportions itself than to the word embedding process. Moreover, as shown in [30], there is a plethora of hyper-parameters which can be tuned in the embedding algorithms, which can have a large impact on the success of word representation methods. Contrary to the initial claim of [43], the superiority of GloVe is debatable: other embeddings, after parameter tuning, perform better than GloVe in a majority of analogy-related tasks [15, 8]. For that reason, GloVe is a good candidate to test our approach because its success could not be attributed to the performance of the embedding.

In short, to increase the performance of a task, we could (a) tune an embedding to suit the formula, or (b) to find or learn another formula to capture the hidden relationships in the given embedding [15]. In this paper, we choose option (b). More precisely, we proceed as follows:

1. We try machine learning algorithms as binary classifiers of analogies. Such a classifier takes as input a quadruple of words (a, b, c, d) and output 1 if $a : b :: c : d$ holds, 0 otherwise.
2. We also test algorithms for solving analogical equations such as $a : b :: c : x$. The solving process is a regression task where we have to approximate a hidden function f such that the solution of the equation $a : b :: c : x$ is $x = f(a, b, c)$.

For experimenting our approach on the two tasks above, we need to get datasets of quadruples of words (a, b, c, d) which are considered as cognitively acceptable analogical proportions. Moreover, if we consider that an analogical proportion $(a : b :: c : d)$ can be built from two pairs of words (a, b) and (c, d) sharing the same relation R, we can also take advantage of dataset of pairs to build new analogical proportions. This means that, starting from two pairs (a, b) and (c, d), which are known to belong to the same class, we may consider $a : b :: c : d$ as a valid analogical proportion. Still, the analogical proportions obtained this way may be debatable as discussed in Subsections 2.4 and 4.1

As a consequence, we experiment on 4 datasets:

1. The first one is the standard Google Analogy Test Set having more or less 20,000 analogical proportions[2].
2. The second one is BATS (Bigger Analogy Test Set)[3] [18], which is made of pairs, and as such departs from Google dataset. From the pairs in BATS, we can obtain 99,200 quadruples of words viewed as analogical proportions in 40 morphological and semantic categories.
3. The third one, usually called DiffVec[4] [59] is also made of pairs. Vylomova et al. [59] have introduced this large dataset covering many well-known lexical relation types, leading to a total of 36 classes of pairs, highly unbalanced in terms of size (from a class of 6 examples to a class of 3,583 examples), with a total number of 12,458 pairs. DiffVec refers to Difference Vectors method. Bouraoui et al. [8] keeps the name DiffVec for the dataset of pairs introduced in [59]. We shall continue to use DiffVec for referring to the set of quadruples that we build by taking pairs of pairs in the same class (as we do with BATS).
4. Finally, the fourth one is SAT (Scholastic Assessment Test), a former college entrance exam [57][5]. This dataset is quite small with only 374 analogies such as $livid : anger :: radiant : happiness$ but it is so challenging that even a very good student will get no more than 60% success rate.

This paper is a fully revised and expanded version of a conference paper [31]. Apart from adding more extensive sets of experiments, we have also investigated in more detail the impact of the formal modeling of analogical proportions on the machine learning implementation. We make several contributions to the field as summarized below:

1. We formally investigate what can be expected from an analogical relation. We show that an intuitive property like unicity postulate does not fit with word analogies. As such, it has to be rejected.
2. Starting from the previous formal properties expected from an analogy relation, we build positive and negative training sets from our datasets. Doing so, we are able to build classifiers from a proper set of examples.

[2]Download from http://download.tensorflow.org/data/questions-words.txt.

[3]Download from https://vecto.space/projects/BATS/.

[4]Download from
https://github.com/ivri/DiffVec/blob/master/word-pairs-final.SEMBLESS.csv.

[5]The SAT dataset has been gracefully made available to us by P. Turney.

3. Apart from testing diverse machine learning classifiers, we design and implement a neural network to decide whether a quadruple of words is an analogy or not, with better accuracy than state of the art algorithms, and even higher accuracy than an average human on the SAT dataset.
4. Finally, we depart from the traditional view of analogical proportions where finding x such that $a : b :: c : x$ holds is based on a parallelogram view where the missing x is just the vertex of the parallelogram (a, b, c, x). The machine learning approaches outperforms state of the art approaches on Google, BATS and DiffVec datasets.

As far as we know, neither analogy classification nor analogy completion have been investigated in the same way as we have proposed in this paper, namely learning a model, instead of starting from the parallelogram model.

The paper is structured as follows. Section 2 recalls the postulates characterizing analogical proportions and identifies a rigorous method for enlarging a set of examples and counter-examples (also known as data augmentation in the machine learning community). Section 3 and its subsections provide a new approach to the recognition/classification of analogical proportions, experimenting with several algorithms (SVM, random forests, neural networks) for this purpose. Moreover, we also propose a learning approach for solving analogical proportion equations in natural language (often referred to as analogy completion in the natural language processing literature). Section 4 presents and discusses the experimental settings and Section 5 reports results for datasets Google, BATS and DiffVec. Experiment results show that machine learning-based techniques are more accurate than the state of the art, both for classification and regression tasks. We dedicate Section 6 to experiments and results on the SAT dataset. Related work is discussed in Section 7, before concluding in Section 8.

2. Analogical proportions: What we can expect

Analogical proportions have a long history and the best way to tackle the issues we want to investigate is to start from a simple formal modeling. From a machine learning perspective, it is also interesting to understand what is the negative example of an analogy.

2.1. Basic postulates

Taking inspiration from the properties of numerical proportions, such as geometric proportions (i.e., $\frac{a}{b} = \frac{c}{d}$), or arithmetic proportions[6] (i.e., $a - b = c - d$), analogical proportions are quaternary relations, supposed to obey the three following first-order logic postulates (e.g., [27]): $\forall a, b, c, d,$[7]

1. $a : b :: a : b$ (*reflexivity*);
2. $a : b :: c : d \to c : d :: a : b$ (*symmetry*);
3. $a : b :: c : d \to a : c :: b : d$ (*central permutation*).

These basic definitions have been widely investigated in [45]. We recall here some of the direct consequences of these postulates like

- $a : a :: b : b$ (*identity*);
- $a : b :: c : d \to b : a :: d : c$ (*internal reversal*);
- $a : b :: c : d \to d : b :: c : a$ (*extreme permutation*);
- $a : b :: c : d \to d : c :: b : a$ (*complete reversal*).

Repeated applications of postulates 2 and 3 show that an analogical proportion has exactly *eight* equivalent forms:

$$\begin{aligned} & a : b :: c : d \\ = & c : d :: a : b \\ = & c : a :: d : b \\ = & d : b :: c : a \\ = & d : c :: b : a \\ = & b : a :: d : c \\ = & b : d :: a : c \\ = & a : c :: b : d. \end{aligned}$$

All these properties strictly fit with the intuitive meaning of a word analogy. Nevertheless, an intuitive property that we call *transitivity*, namely:

$$(a : b :: c : d) \land (c : d :: e : f) \to a : b :: e : f$$

[6]Note that the parallelogram view of analogical proportions, defined by $\overrightarrow{ab} = \overrightarrow{cd}$, is a point-wise extension of the arithmetical proportion.

[7]In the following of this section, we will omit the universal quantifier for readability.

is not derivable from the axioms [45]. Despite this property is valid when we consider, for instance, numerical, arithmetical ($a - b = c - d$) or geometrical ($a \times d = b \times c$) proportions, it is not universally valid in the sense that we can find a model of analogy which does not satisfy this property.

Considering words in natural language, if everybody agrees that $nurse : patient :: mother : baby$ and that $mother : baby :: frog : tadpole$ should hold, very few people will consider $nurse : patient :: frog : tadpole$ [11] as a valid analogy. In some sense, it is satisfactory that postulates 1 - 2 - 3 do not allow the derivation of transitivity.

2.2. Unicity postulate

All the three postulates of analogical proportions are intuitively satisfactory, and we could ask for something a little bit stronger than postulate (1), but still not implying transitivity:

4. $\forall a, b, x,\ a : b :: a : x \to (x = b)\ \ (unicity)$

Consequences of postulates 2 - 3 - 4 that can be derived.

Fact 1. $a : a :: b : x \implies x = b$

Proof: $a : a :: b : x \implies a : b :: a : x$ (by 3) $\implies x = b$ (by 4) □

Fact 2. $a : b :: c : c \implies a = b$

Proof: $a : b :: c : c \implies c : c :: a : b$ (by 2) $\implies a = b$ (by Fact 1) □

Fact 3. $a : b :: c : b \implies a = c$

Proof: $a{:}b{::}c{:}b \implies a{:}c{::}b{:}b$ (by 3) $\implies a = c$ (by Fact 2) □

Fact 4. $a : b :: c : d \wedge (a \neq b) \implies c \neq d$

Proof: Suppose $c = d$. Then $a : b :: c : d$ is $a : b :: c : c$, and by symmetry, $c : c :: a : b$ which implies $a = b$ (by Fact 1). Which contradicts the hypothesis $a \neq b$. □

Clearly, all the above properties are satisfied by arithmetic proportions. But, when it comes to word analogies, postulate (4) is not satisfied: for instance, $orange : orange :: banana : yellow$ is still a valid word analogy that does not match Fact 1. In fact, all homonym words like *book, project, close, well, etc.* could be likely involved in valid analogical proportions of the type $a : a :: c : d$ involving exactly three distinct words. So, in the following, we do not assume postulate 4.

Besides, in [41], the authors, acknowledging the fact that human biases can be found in word embeddings (leading to validating analogical proportions such that "man is to computer programmer as woman is to homemaker"), discuss analogical proportions such as "man is to doctor as woman is to doctor". As can be seen, this proportion violates Fact 3, a consequence of unicity postulate 4, without involving homonym words. This later proportion expresses than in the context 'doctor' there is no difference between 'man' and 'woman'. Such proportions have an argumentative flavor, since it expresses what *should be*, rather than a current state of things as acknowledged by language corpora. Another use of proportions may be explanatory as in "the denarius is to the church as water is to life", suggesting that 'denarius' is something indispensable for the life of the church. In such case, even the symmetry of (a, b) and (c, d) becomes debatable. Dealing with such analogical proportions is beyond the scope of this paper.

2.3. Analogy classes and their underlying structure

Given four distinct items a, b, c, d, they can be ordered in $4! = 24$ different ways. Among these 24 permutations of 4 distinct items, there are 3 classes of 8 permutations, each one being stable under the postulates of analogical proportions, since $a : b :: c : d$ can be written in 8 equivalent forms.

As a consequence, as soon as a, b, c, d are all distinct, $b : a :: c : d$ and $a : d :: c : b$ do not belong to the same class as $a : b :: c : d$ and are in fact elements of two other different classes. If an element of a class is a valid (resp. not valid) proportion, then the seven remaining ones are also valid (resp. not valid). Although it does not follow from postulates 2 - 3 - 4, one can also consider that if $a : b :: c : d$ holds then neither $b : a :: c : d$ nor $a : d :: c : b$ hold as valid analogical proportions. This can be observed on the example $a =$calf, $b =$cow, $c =$foal, $d =$mare.

It has to be noted that, if $a : b :: c : d$ holds, then there is no permutation among the eight equivalent configurations displayed above that is of the form $b : c :: x : y$, while there are configurations of the forms $a : b :: x : y$ and $a : c :: x : y$ (and all their transforms by symmetry and central permutation). Concerning the binary relation interpretation, it suggests that the link between b and c is not relevant for the proportion to hold, while the links between a and b and between a and c are relevant. This will be useful when it comes to building a neural network for regression.

Despite their obvious semantic, these fundamental properties of analogical proportions are rarely used in practice. However, they have implications when it comes to machine learning, as we will see in Section 4.

2.4. Other types of analogical proportions

As pointed out in [2], we may distinguish between two types of word analogy. One type where the four words belong to the same *conceptual space*, as in, e.g., `"calf"`:`"cow"`::`"foal"`:`"mare"` (where all the words refer to animals), while the other type involves two distinct *conceptual spaces*, as in `"wine"`: `"French"`::`"beer"`:`"English"` (where the words refer to drinks and people)[8]. For this second type, it appears that *central permutation* may not be applicable: only *symmetry* and *internal reversal* seem to be acceptable. In such a case, an analogical proportion would have only *four* equivalent forms, as shown below

$$a : b :: c : d = c : d :: a : b = b : a :: d : c = d : c :: b : a$$

Note that *symmetry* and *internal reversal* entail *complete reversal*, which looks acceptable for analogical proportions involving two conceptual spaces.

As we understand, distinguishing the two types of analogical proportions might have consequences not only on the logical modeling but also on its experimental counterpart. However, there are also examples of analogical proportions involving two conceptual spaces where central permutation is tolerable, for instance with:

`"Tokyo"`: `"Japan"`::`"Paris"`:`"France"`

This proportion is based on the relation $ToBeTheCapitalOf$ shared by the two pairs. For a long time, there have been many discussions about what makes an analogical proportion valid or fallacious [21, 50]. Generally speaking, it may seem natural to assume that an analogical proportion can simply be defined as:

$a : b :: c : d$ iff for some relation R, $R(a, b)$ and $R(c, d)$ hold.

This assumption is implicit when using datasets such as BATS and DiffVec. This definition, which is clearly symmetrical, has several consequences:

- The following implication should hold $a : b :: c : d \implies b : a :: c : d$ just because $R^{-1}(a, b)$ and $R^{-1}(c, d)$ hold.

[8]Note that $orange : orange :: banana : yellow$ is another example, while `"wine"`:`"French"`::`"beer"`:`"x"` also fails to satisfy *unicity* since, e.g., `"x=Belgians"` would fit as well.

- It is unclear that the fact that $R(a, b)$ and $R(c, d)$ hold always entails that $S(a, c)$ and $S(b, d)$ hold for some relation S (beyond general relations, such that S = "being in the same category" i.e. `"Tokyo"` and `"Paris"` are both capital cities). See [21] for a discussion. This questions the validity of central permutation for relation-based analogical proportions. Think, e.g.,

  `"Tokyo": "Japan"::"Paris":"France"`

- If R is symmetrical, i.e., $R(a, b) \implies R(b, a)$, then

  $$a : b :: c : d \implies b : a :: c : d$$

  which is quite debatable as shown by the following example:

  `"black":"white"::"white":"black"`

  that cannot be considered as a satisfactory analogical proportion despite the fact that the binary symmetrical relation $IsOppositeOf$ is satisfied by both sides.

- The relation R may be arbitrary, quite general or vague, as in:

  `"camera":"water"::"attention":"baby"`

  which would hold because the number of syllables of the first word in a pair is just the number of syllables of the second word increased by 1.

In summary, the only properties that hold for sure with relation-based analogical proportions are 'symmetry', 'internal reversal', and 'complete reversal', while other properties are debatable. Ultimately, the acceptable orderings of 4 words making a proper analogical proportion may be viewed as a matter of convention, even if some orderings are certainly more cognitively valid than others. This is not an issue for the machine learning approaches proposed in this paper since they consider as valid analogical proportions whatever is given in the dataset of positive examples. So there is no harm in applying the 8 permutations to each quadruple issued from these datasets: either the quadruple is an analogical proportion satisfying all the postulates, or it is a "weak" one and the result will be another "weak" one. Indeed, in this work, we are going to use benchmarks made of analogical proportions of various qualities. This is the case with datasets originally made of pairs of words, classified into categories.

3. Analogy classification and completion: the proposed approaches

We take advantage of the previous theoretical analysis for revisiting the problems of identifying and solving analogical proportions expressed in natural language. We propose a new approach to these problems, which have been widely investigated by the NLP community [36, 55, 27, 28, 29]. It is now common to convert words into numerical vectors for computational purposes, a process known as word embedding. If V is the target vector space and W the corpus of words, we denote $embed(W)$ the subset of V representing the words of W, and obviously, whatever the embedding process, $embed(W) \subset V$. The underlying assumption is that a "good" word embedding encodes linguistic relations in such a way that they are identifiable via linear vector offset. This leads to the well-known parallelogram view of analogies. In fact, this view has to be seriously tuned to detect analogies in a diverse corpus of data [18, 30, 8]. Given the embedding process (here GloVe in the experiments), we depart from this assumption and tackle the issue from another viewpoint. Instead of assuming that a word analogy $a : b :: c : d$ is valid iff $embed(a) - embed(b) \simeq embed(c) - embed(d)$, which allows to classify a quadruple of words, we learn from word analogies examples how to classify without assuming any underlying predefined formula. Equipped with the observations from Section 2, we have a rigorous way to augment the initial datasets, as it is often needed in machine learning. Let us first investigate some works related to what we do and which could, in one way or another, serve as baselines, before presenting our approaches to analogy classification and completion.

3.1. Analogy classification and completion in the literature

The method for deciding if four items a, b, c, d constitute an analogical proportion depends on their representation level. In case a, b, c, d are described in terms of Boolean, nominal or numerical features, this binary classification problem can be directly solved from the definitions of $a : b :: c : d$ for these different kinds of features [45], [16].

In case a, b, c, d are words represented by their embeddings (the case under interest in this paper), the representation relies on the parallelogram view. From a strict vector offset interpretation, it amounts to state that $a : b :: c : d$ holds iff $a - b = c - d$.[9] Allen and Hospedales [10] enable word analogies to be probabilistically grounded, providing the first rigorous explanation for the presence of

[9]For the sake of simplicity, we use the same notation for a word and its embedding.

linear relationships between the word embeddings of analogies: this could explain why the parallelogram relation is often satisfied.

Analogy completion is the task of looking for x such that $a : b :: c : x$ holds. The method for solving the problem (and its difficulty) again depends on the representation level used for a, b, c. In case a, b, c are described in terms of Boolean, nominal or numerical features, the computation of the representation of the solution is relatively easy; see [5, 12]. When words are represented by embeddings, the parallelogram view leads to computing $c - a + b$ as a good approximation of the embedding of the solution d. It has been recognized that the straightforward computation of d as being the nearest neighbor of $c-a+b$ in terms of cosine similarity (namely considering $argmax_{d \in embed(W)} cos(d, c - a + b)$ as the solution) is not fully satisfactory in practice.

- That is why in [29], Levy and Goldberg introduced the $3CosMul$ formula as a replacement of the above additive formula. We may notice that (1) is not far from a geometric proportion-based view of analogical proportions as $\frac{a}{b} = \frac{c}{d}$ (which would lead to $d = \frac{b \times c}{a}$):

$$argmax_{d \in embed(W)} \frac{d \cdot b \times d \cdot c}{d \cdot a + \epsilon} \tag{1}$$

  where $\cdot$ is the scalar product. $3CosMul$ finds the final solution d by ranking all words in the dictionary according to the above quotient, and choosing the word d which generates the highest value.

- Another parallelogram variant is $argmax_{d \in embed(W)} cos(d - c, b - a)$ (named PairDirection in [29] and PairDistance in Drozd et al. [15]). One may also use a weighted mean of $cos(d - c, b - a)$ and $cos(d, c - a + b)$ as in [25]. Improving PairDistance, the authors of [15] introduced the $LRCos$ method. This is an alternative approach starting from a set of word pairs (a, b). This set is organized into subsets involving pairs sharing a common relation. In this context, $a : b :: c : d$ is understood as $a\mathcal{R}b$ and $c\mathcal{R}d$, for some relation $\mathcal{R}$ where a and c are source words, b and d are target words. For instance 'Japan is to Tokyo as France is to Paris' written as $Japan : Tokyo :: France : Paris$, and $\mathcal{R}$ is the 'country-capital' relation.

  Given a set of word pairs $(source, target)$, then "the available target words are used as positive samples, and source words, along with random words from the dictionary, as negative samples" in a logistic regression problem. This means, continuing the same example with a given set of valid pairs

$(Japan, Tokyo)$, $(France, Paris)$, $(Australia, Canberra)$, etc., positive examples contain $Tokyo, Paris, Canberra$, etc., negative examples contain $Japan, France$, $Australia$, etc. plus some randomly chosen words from the dictionary, not belonging to the 'capital' class. When given a test word such as $England$, $LRCos$ tries to find words in the class 'capital' and close to $England$.

- The works of Bouraoui et al. [8] are in a relation induction perspective. Having a set of pair examples (a, b) satisfying some relation, the authors of [8] are looking for pairs (c, d) satisfying the same relation. This is not exactly what has been done in [15], where a, b, c are given and the authors focus on analogy completion.

  Bouraoui et al. suggest two models for identifying word pairs: a translation model and regression model. While the translation model relies on the difference between the embedding of the 2 words $a - b$, thus assuming a relation like $b = a + v$, the regression model assumes a more general linear relation such as $b = U.a + v$ (where U is a matrix, b, a and v are vectors or the same dimension). In the translation model (in the spirit of $LRCos$), the authors want "to accept (s, t) as a valid instance if (i) s and t are sufficiently similar to the vector representations of the given source and target words, and (ii) the translation $t - s$ has a sufficiently high probability". The translation model departs from $LRCos$ since in the analogy completion task of [15], s is always given as a valid source word while in a relation induction problem, the authors "need to consider the probability that s is a valid 'source word' ". The regression model relaxes the simplistic assumption of the difference-based approach from the parallelogram view, but is generally outperformed by the translation model. None of these two approaches are used in an analogy completion perspective.

In view of the above descriptions, it is natural to compare our works with:

- The current best method from the translation (and regression) model [8] for analogy classification task, since it outperforms the other approaches;
- $3CosMul$ and $LRCos$ for analogy completion task.

In fact, none of the previous works have been experimented on SAT dataset. As a consequence, regarding our experiences with SAT, we will compare with the works of [56]. We now describe our approach for classifying word analogies, and then for analogy completion.

3.2. Analogy classification

In the works previously reviewed, analogy classification resorts to relation induction. But the problem of deciding if a quadruple of words is a valid analogical proportion or not is in fact a purely binary classification machine learning task. As far as we know, it has not been widely investigated as such; see [54, 3] for other approaches.

We may think of diverse machine learning approaches for analogical proportions classification. We have tried four methods: SVM (Support Vector Machines), Random Forest, Neural Network and Convolutional Neural Network as explained below.

- SVM [13] is a popular classification algorithm[10]. We tried several kernels: linear, rbf, sigmoid and polynomial kernel and weight 1, 2, 3 and 4. While we get poor results with linear kernel, the polynomial kernel, which is computationally expensive, provides excellent results with a weight of 2.

- Random forests [9] have been successfully used for classification in different domains, so it seems to be an appropriate method for comparison. We tried random forest using various parameters, such as the number of trees, maximum depth and split. The best results were obtained with the following parameters:100 trees, no maximum depth, and a minimum split of 2.

- The traditional fully connected Neural Network (NN) is another popular classifier which has been successfully applied in diverse domains. It is interesting to see how well such a network performs compared to the more complex Convolutional Neural Network described in the next section. The network we tried is made up of five fully connected layers. The first four layers use Relu activation, batch normalization with the following numbers of neurons: the first layer has $4 \times n$ neurons, where n is the vector length of the embedding, and the number of neurons is halved for the subsequent layer, so the fourth layer has $n/2$ neurons. In this paper, $n \in \{50, 100, 200, 300\}$. The last layer (the fifth) has one neuron with a sigmoid activation function, The network was trained using the Adam optimization algorithm [26].

[10]For SVM and Random Forest, we used the implementation as provided by `Scikit-Learn` library (https://scikit-learn.org). For NN and CNN, we use standard Keras and Tensorflow libraries.

- Convolutional Neural Network. This network is more complex and is described in more detail in the next section.

3.3. Convolutional Neural Networks for analogy classification

One of the most successful methods for image classification is Convolutional Neural Network (CNN) (see e.g. [40]): this type of network can capture high-level features not easily extracted otherwise, especially within pictures. So, we believe CNN can also capture hidden semantic links underlying a valid analogy by transforming the analogy classification task into an image classification task.

Stacking together the 4 vectors with n components corresponding to a quadruple a, b, c, d of words, we get a matrix $n \times 4$ that we are going to process as we would do for an *image*. With filters respecting the boundaries of the 2 pairs, this is the structure of the CNN:

- 1st layer (convolutional): 128 filters of size $height \times width = 1 \times 2$ with strides $(1, 2)$ and Relu activation. This means that we are working component by component (with a vertical stride of 1) and we move from pair (a, b) to pair (c, d) with horizontal stride 2.

- 2nd layer (convolutional): 64 filters of size $(2, 2)$ with strides $(2, 2)$ and Relu activation. This is an intermediary step for reducing the dimension before going to the final dense layer.

- 3rd layer (dense): one output and sigmoid activation as we want a score between 0 and 1, values of 0.5 or above are considered positive (the default approach for binary classification).

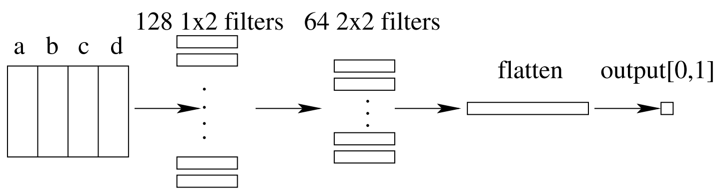

Figure 1: Structure of the CNN as a classifier.

Note that we use only one intermediary layer due to the modest size of the input. The first two convolution layers use batch normalization and Relu activation function, the output layer uses sigmoid activation. The CNN was trained using Adam optimization algorithm. This is a standard way to proceed. The structure of this network can be seen in Figure 1. Note that the CNN approach returns a number between 0 and 1 which might be considered as an empirical estimate of the quality of the candidate analogical proportion. However, the merit of this estimate has not been further empirically explored.

In Section 5, we show the results for Google (Table 4), BATS (Table 5) and DiffVec (Table 6). In Section 6, we provide results for SAT. These tables also report the baseline results to get a clear comparison. Some more detailed tables can be found in annexes.

3.4. Analogy completion

In this paper, a, b, c are words represented by their respective embeddings, not even assuming any information about some categorization of the words. We agree that the solution d to the analogical equation $a : b :: c : x$ may be a function f of a, b, and c. But, in contrast to the works described in subsection 3.1, we neither assume a given formula for f nor consider the completion task from a relation induction viewpoint. We propose to solve the equation $a : b :: c : x$ by learning x via regression. This is a multi-variable regression problem where the training set is built from the initial set of analogies $a : b :: c : d$ as pair $((a, b, c), d)$. This does not necessarily call for a functional view, still neural net approaches fit with such a view. For predicting x from a given triple of words (a, b, c) and a training set of complete analogical proportions, can be achieved via various machine learning techniques.

We may think of using statistical methods regression which do not refer to some hidden function f as it is the case for the parallelogram-inspired methods. We thus tried lasso regression [52, 46] with various parameters, but the results were very poor: the best accuracy we got is 42.8%, without normalisation and with an alpha value of 0.001.

From another viewpoint, analogy completion may be considered as a matter of estimating a hidden function f such that $f(a, b, c) = d$ where d is the solution of the completion problem $a : b :: c : x$. Considering a multi-layer neural network as a universal function estimator, it is appropriate to use such a network to estimate our target function. Regarding the equation solving problem, we first consider a, b, c as the input of our network, and d as the output.

We first designed a neural net with $3 \times n$ input neurons and n output neurons. We got very poor results. We think that this initial failure was due to the way we process the 3 inputs without taking into account the similarity/dissimilarity: between a and b on one side and between a and c on the other side. So we move

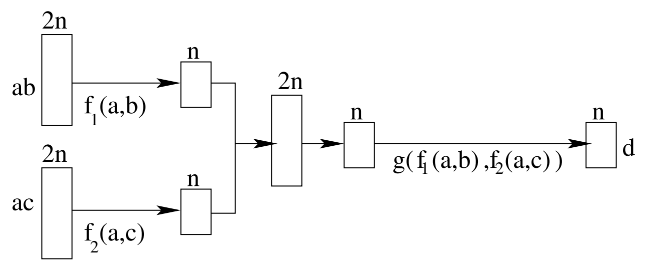

Figure 2: Structure of the neural network for analogy completion.

to a more sophisticated network whose structure parallels the following hidden links:

1. The link between a and b is described via a (unknown) function f_1;
2. The link between a and c is described via a (unknown) function f_2;
3. The final function g to be estimated takes $f_1(a, b)$ and $f_2(a, c)$ as parameters i.e.: $d = g(f_1(a, b), f_2(a, c))$.

As described in Section 2.3, the link between b and c is irrelevant for the proportion to hold, so there is no need to establish this relationship in the network. Using this remark, we approximate this function $g(f_1, f_2)$ via two networks approximating f_1 (resp. f_2). A final neural network approximating g received as input two values: the output of f_1 and the output of f_2. The output of $g(f_1, f_2)$ is d, is then an approximation of $d = g(f_1(a, b), f_2(a, c))$. Figure 2 describes the structure of the regression neural network.

The first layer takes as its input (a, b) and (a, c), effectively estimating the functions $f_1(a, b)$ and $f_2(a, c)$. The input dimension of this first layer for these two functions is twice the length of the input; that is, if the dimension of a, b or c is 50, then the input is 100. In the second layer, the output from the first layer of these two functions is reduced to dimension n; the second layer effectively acts as an encoder. The third layer concatenates the output of the second layer. The fourth layer (the last hidden layer) also acts as an encoder by reducing the size of the dimension to n before producing the final results (d) which also has n dimension. All hidden layers use Relu activation function, the last layer (output) uses linear activation function. The network was still trained using Adam optimization algorithm. The output of the network, d', is unlikely to be a GloVe embedding, so we have to find the word d the most similar (in terms of cosine similarity) to d', i.e. the nearest neighbor of d' in $embed(W)$. This word d is hopefully the correct answer.

4. Experimental settings

As explained in Section 1, in all our experiments, we use GloVe [43] as the embedding model. Glove provides several publicly available pre-trained $\mathbb{R}^n$ word vectors which can be downloaded from https://nlp.stanford.edu/projects/glove/, with dimension n in $\{50, 100, 200, 300\}$ (Wikipedia dump). Ultimately, our datasets are loaded as CSV files. In the case of our CNN for classification, each row is compiled into a real-valued matrix of dimension $n \times 4$ that we consider as an image. The GloVe embedding originally contained 400,000 words. To speed up processing time, we removed non-alphabetical words (numbers, punctuations, underscores), and we are left with 317,544 words. This has no effect on the embedding process as the removed words do not appear in the analogy datasets[11]

Before presenting our four datasets (Google, BATS, DiffVec and SAT) in detail, and the way we build a training set for machine purposes, we discuss the link between relational induction and analogy classification, and we propose a measure of the quality of a dataset as a repository of analogical proportions, given a word embedding.

4.1. Link with relation induction

In the NLP literature, Relational Induction (RI) is often linked to building analogies. In that context, RI is looking for the relationship (e.g. antonymy, synonymy, category membership, etc.) between two words: this can also be considered as a classification task for pairs of words where the relation between two words is the class label. BATS [18, 15] and DiffVec [59] are two well-known datasets dedicated to RI experiments. Let us consider 3 examples obtained from DiffVec [59] by combining two pairs from the same class:

- $(accident, damage)$ and $(bath, cleanliness)$ are two instances of a relation expressing a link $CAUSE - PURPOSE\ Cause : Effect$. Joining these two instances lead to a debatable analogical proportion:

  $$accident : damage :: bath : cleanliness$$

  The cognitive difficulty when looking at this quadruple as an analogical proportion seems to come from the fact that causality is quite a general relation, and that here the physical mechanisms at work in accidents and baths have little in common.

[11]. Our whole python code with the complete requirements, as well as our datasets (except SAT) are publicly available on github repository https://github.com/gillesirit/analogy.

- Looking at class $CAUSE - PURPOSE$ $Action/Activity : Goal$, from which we obtain:

  $$speak : express :: starving : hungry$$

  which looks weak for reasons similar to the previous example.

- The relation $ATTRIBUTE$ $Action : ObjectAttribute$ is satisfied by the two word pairs $(paint, house)$ and $(wine, drink)$. Still, it is hard to think of

  $$paint : house :: wine : drink$$

  as a genuine analogical proportion, since the explanation to support it seems quite intricate.

The parallel between two pairs sharing some relation does not always seem to be a sufficient requirement for making widely acknowledged and thus cognitively valid analogical proportions, as discussed in detail in Subsection 2.4. The above remarks tend to suggest that, strictly speaking, both BATS and DiffVec cannot be considered as proper analogical proportions datasets. A measure of the analogical quality of a dataset, supporting this claim, is presented in the next Subsection. However, as mentioned before, the proposed machine learning procedures still apply to "weak" analogical proportions.

Let us point out that RI task is not the same, strictly speaking, as analogy classification. Indeed RI amounts to classifying new pairs among a set of candidate relations (e. g., 36 classes for DiffVec and 40 for BATS). Our analogy classification task deals with quadruples (positive and negatives examples built from pairs of the same class when dealing with BATS and DiffVec) and is a binary classification problem: deciding if a quadruple of words constitutes, or not, an analogical proportion.

Moreover, generally speaking, the task of classifying a quadruple (a, b, c, d) as a valid proportion does not necessarily require identifying the relation linking a and b (or even (c, d)).

Nevertheless, a proper analogy classifier could be used for RI in the following way: Having a finite set of classes and knowing that pair (a, b) belongs to class C, checking the quadruple (a, b, c, d) with the analogy classifier could tell us if (c, d) also belongs to C or not. However, this is not the purpose of this paper to apply analogy classifiers to RI.

4.2. Average analogical dissimilarity of a dataset

As suggested by Linzen [32], the parallelogram equation for solving word analogies has become a standard evaluation tool for words embedding (such as GloVe).

In the following, we reuse this idea in a different perspective. We thus assume that given a reasonably good word embedding, such as GloVe, datasets made of high quality analogical proportions should lead to parallelograms of better quality than datasets containing weak proportions.

Let us explain how we have checked this assumption on the 4 datasets. If $a : b :: c : d$ is considered as a valid analogy, then

$$embed(a) - embed(b) \simeq embed(c) - embed(d)$$

or equivalently

$$||embed(a) - embed(b) - embed(c) + embed(d)|| \simeq 0$$

So, as proposed by Miclet et al. [35], the so called *analogical dissimilarity*

$$AD(a, b, c, d) = ||embed(a) - embed(b) - embed(c) + embed(d)||$$

could be considered as a good measure of how ”far” $a : b :: c : d$ is from being an analogical proportion, given the embedding. In other words, when $AD(a, b, c, d)$ is close to 0, $a : b :: c : d$ should be read “a is to b almost as c is to d”. We have then computed this analogical dissimilarity for each quadruple of the 4 datasets we have used, then getting the average value per dataset. The results are shown in Table 1. Note that the values are not normalized w.r.t. the dimension. Clearly, this average analogical dissimilarity also captures the imperfect nature of the word embedding. As expected, Google dataset has the smallest deviation, followed

dataset	dim 50	dim 100	dim 200	dim 300
GOOGLE	3.86	4.8	6.07	6.79
BATS	5.55	6.55	8.13	9.01
DIFFVEC	6.42	7.6	9.21	10.05
SAT	6.42	7.47	9.26	10.2

Table 1: Datasets average AD for GloVe dimensions 50, 100, 200 and 300

by BATS, DiffVec and SAT. Surprisingly, the deviations of SAT and DiffVec are very similar, but SAT is regarded as a repertory of high quality proportions. This is likely due to the fact that SAT proportions involve sophisticated vocabularies, which occur less frequently in the corpus documents so the models for those words may be less accurate.

4.3. Datasets

Experiments have been performed with 4 distinct datasets:

1. The one proposed by Mikolov et al. [37] - from Google (questions-words file), containing exactly 19,544 analogies, each line has four distinct words. These analogies are classified in 14 categories such as 'common-capitals, countries' like $Athens : Greece : Ottawa : Canada$, 'country-currency', or 'opposite' like $acceptable : unacceptable :: aware : unaware$, etc. It appears that some analogy can appear in different categories and some of the categories may contain several occurrences of the same quadruple of words (but in a different order). When we removed these redundancies, we got a final dataset of 12,771 distinct analogies. Table 2 describes the classes with examples, their cardinality with and without redundancy.
2. The one proposed by Gladkova et al. [18, 15], BATS. In fact, BATS is a set of words pairs (a, b), like $(bat, cave)$. But since $(bat, cage)$ is also acceptable, BATS contains lines such as $(bat, cave/cage)$. All in all, BATS allows to build 99,200 analogies from such extended pairs. Due to space limitation, we omit the table of 40 rows showing BATS categories and denote them as

$$E01, \dots, E10, D01, \dots, D10, I01, \dots, I10, L01, \dots, L10.$$

   However, for the interested reader, this complete table can be found in [15].
3. The one proposed by Vylomova et al. in [59], DiffVec, initially built of pairs (a, b) with their corresponding class that we have transformed into a set of analogical proportions just by considering all options $(a : b :: c : d)$ as soon as (a, b) and (c, d) belong to the same class (among 36).
4. A dataset of 374 analogies coming from the SAT college entrance test [12]. The analogies in this dataset are entirely semantic and are not syntactic (i.e. not morphological). That is why dealing with this dataset is highly challenging. As some analogies make use of words without GloVe embedding, we ultimately end up with 367 analogies having their GloVe counterpart.

[12]This dataset has been kindly provided to us by Peter Turney.

	a	b	n	$no\ redundancy$
$Common\ capitals$	$athens$	$greece$	506	253
$All\ capitals$	$abuja$	$nigeria$	$4,524$	$4,353$
$US\ cities$	$chicago$	$illinois$	$2,467$	2121
$Currencies$	$algeria$	$dinar$	866	433
$Nationalities$	$albania$	$albanian$	$1,599$	820
$Gender$	boy	$girl$	506	253
$Plurals$	$banana$	$bananas$	$1,332$	666
$Base\ to\ gerund$	$code$	$coding$	$1,056$	528
$Gerund\ to\ past$	$dancing$	$danced$	$1,560$	780
$Base\ to\ 3^{rd}\ person$	$decrease$	$decreases$	870	435
$Adj.\ to\ adverb$	$amazing$	$amazingly$	992	496
$Adj.\ to\ comparative$	bad	$worse$	$1,332$	666
$Adj.\ to\ superlative$	bad	$worst$	$1,122$	561
$Adj.\ un-prefixation$	$acceptable$	$unacceptable$	812	406
$Total$			$19,544$	$12,771$

Table 2: Analogies categories for Google dataset

4.4. Extending datasets for classification

Due to the properties of analogical proportions reviewed in Section 2, it is easy to rigorously extend a dataset of analogies by applying the diverse postulates. This process is commonly known as data augmentation, since one uses properties or structures of the specific data available to produce new ones. Although the most common application is in image processing (where results should be invariant by rotation, zoom, etc.), it can also be applied to analogies using their structural properties. Starting from an initial dataset of n distinct analogies, each one involving 4 different words, we end up with $8 \times n$ valid analogies which constitute the positive examples. But in machine learning, negative examples are also needed.

By using the same initial dataset, we get invalid analogies just by permuting 2 elements of a valid analogy $a : b :: c : d$:

- Starting from n valid analogies and permuting the 1st and the 2nd elements provides an invalid analogy. Then permuting 8 times this invalid analogy, we still get 8 invalid analogies leading to $8 \times n$ invalid analogies.

- Then, applying the same process after permuting the 1st and the 3rd elements, leading again to $8 \times n$ invalid analogies.

We get a final extended dataset of $n \times 24$ examples, among which $n \times 8$ are positive examples and the remaining $n \times 16$ are negative examples.

Note that we apply central permutation in a systematic way, although we pointed out in Subsection 2.4 that it may be debatable. However it does not affect the results as shown in experiments; indeed we never test both (a, b, c, d) and (a, c, b, d). Another important point to note: the negative examples are only permutations of the positive ones. As it is highly unlikely that 4 random words (a, b, c, d) constitute a valid analogy, we have also experimented by adding such random negative examples. But this random extension did not bring better results than the logical extension. This should not come as a surprise: adding negative examples involving the same words as the positive examples is likely to be much more informative than adding random negative examples, since the negative examples obtained by permutation are "closer" to the positive ones.

In the following, we denote a dataset of n analogies by TS and the extended dataset of $24 \times n$ elements by $Ext(TS)$. We have diverse ways to use our extended datasets as explained below.

4.5. Working with extended datasets for classification

From a machine learning perspective, we have to split a dataset TS into 2 subsets $train(TS)$ and $test(TS)$. Let us denote $split(S)$ a function leading to a pair of disjoint subsets $train(S), test(S)$ from a set S. It is quite clear that creating a pair of training and testing set, starting from an initial dataset TS can be done in three ways:

1. **Method 1**: Either we start from the full dataset TS at hand, extend this set then split into train and test subsets, i.e. we have to implement

$$split(Ext(TS)) = (train(Ext(TS)), test(Ext(TS)))$$

2. **Method 2**: Or we first split the dataset TS at hand into two subsets and extend them leading to the pair

$$Ext(split(TS)) = (Ext(train(TS)), Ext(test(TS)))$$

3. **Method 3**: Or we split the dataset TS but extend only the training set. This means that for classification, the test data has only positive class:

$$(Ext(train(TS)), test(TS))$$

Method 1 and Method 2 have exactly the same number of examples in total. This is not the case for Method 3 as there are fewer elements to be tested. Intuitively, the performances from these methods may be different: Method 2 and Method 3 can be considered tougher in the sense that there is no permutation of any training analogy in the testing set: the training and testing sets are entirely disconnected.

4.6. Experimental comparison between Method 1, Method 2 and Method 3

Before launching a full batch of experiments which are very time consuming, we have decided to investigate if one method is definitely better than the other. We consider the Google dataset as clean and simple for analogy classification task, so we started our experiments with it in order to get some indication about the most relevant method to use.

Using the CNN described in Section 3, we experimented in dimension 50, with a split ratio train/test of 0.2, i.e 20% of the dataset are retained for testing, the remaining 80% used for training. The resulting accuracies and losses (cross-entropy) are in Table 3. As we can observe, there is no clear advantage

dim 50	$Method\,1$	$Method\,2$	$Method\,3$
5 *epoch*	93.7% ($loss$: 0.155)	94.2% ($loss$: 0.150)	93.05% ($loss$: 0.186)
10 *epoch*	95.0% ($loss$: 0.131)	94.9% ($loss$: 0.125)	92.63% ($loss$: 0.198)

Table 3: Comparison Method 1 - Method 2 - Method 3 with accuracy and loss

of one method over another (at least with these parameters). As a consequence of these indicative experiments, we choose Method 2 in the following, because we consider this method provides a clean separation between training and testing sets (which is not the case with Method 1) and generates a bigger test set than Method 3.

5. Experimental results for Google, BATS and DiffVec

This section is devoted to the results on the classification and regression tasks for datasets Google, BATS and DiffVec; Section 6 covers the SAT dataset. Note that for the DiffVec dataset, we exclude classes larger than 62,000 analogies, as some algorithms require more resources than what we have: Classes 13 and 36 are excluded for this reason. For SVM, we also exclude class 14 and 16 as they are too big for SVM. For LRCos, we also exclude Class 1 and Class 15, as there are only six word pairs and LRCos requires more samples.

5.1. Analogy testing as a classification task

We employ the commonly used metrics for classification tasks: precision, recall, F1, and accuracy collected from 10-fold cross-validation. A summary of the results showing the performance of the SVM, Random Forest, Neural Network (NN) and CNN classifiers is given in Table 4 for Google dataset, in Table 5 for BATS and in Table 6 for DiffVec. Complementary tables are provided in annexes. The NN was trained using 10, 30, 100, 200, 300 and 400 epochs. We present the results for 30 and 100 epochs just because there are no big changes for more than 100 epochs. The CNN was trained using 3, 5 and 10 epochs and we present the results for 10 epochs.

Tables 4, 5 and 6 show that the results are uniformly very good for Google, BATS and DiffVec, whatever the machine learning algorithm used. For Google dataset (Table 4), the accuracies for all four classifiers are nearly 100%. Note that simpler classifiers such as Random Forest and simple Neural Network still achieve very high accuracy even at dimension 100, and the performance is about the same in higher dimensions. For CNN, the accuracy increases by about 2% when the number of dimension increases from 50 to 100. However, the gain appears to be less significant for the higher dimensions, as the accuracy for 200 dimensions is only 0.42% higher than 100 dimensions, and the accuracy for 300 dimensions is actually slightly lower (0.16% lower) than 200 dimensions. It appears that using dimension 100 is more than enough to get a clear view of the value of the 4 metrics. This is why in Tables 5 and 6 we only use at most dimension 100. For BATS, we provide average values for the metrics per BATS class and for DiffVec, a sample of four classes and an average of all classes.

As can be seen in Table 5, the performance of the algorithms vary with the classes. For instance, for BATS, the accuracy of CNN decreases to 69.81% on the Lexicographic-semantics subcategory 'antonyms-binary' (see Table B.14). This should not come as a surprise since the antonym relationship is a very abstract relation and some instances may be even debatable analogical proportions for people, e.g., 'before:after::dive:emerge'.

Table 6 shows the individual results of 4 diverse categories for DiffVec and the average of 34 classes; for the complete results please see Appendix C. For the DiffVec dataset, NN and Random Forest perform the best, followed by CNN, and lastly SVM.

We might think of comparing these results with those of [8]. However, in [8], analogy classification is viewed in terms of relation induction, i.e., deciding if a pair of words (c, d) shares or does not share the same relation with a set of pairs (a_i, b_i). In the case of positive answer, then for all i, $a_i : b_i :: c : d$ is considered as a valid analogy. But, as explained at the end of Subsection 4.1, relational induction is quite a different task: a strict comparison would not make sense.

Note that Bouraoui et al. [8], apart from testing their own models, also tried SVM using a linear kernel. Let us also mention the works of [59] who use a RBF-kernel (Radial Basis Function) SVM classifiers. Our experiments show that we only get good results with a polynomial kernel, which is more computationally expensive compared to a linear kernel. A linear kernel accepts training and testing data as is (without any transformation). However, a polynomial kernel has to transform training and testing data into a higher dimension. The results can be found in Tables 4, 5 and 6: they are quite good.

Dim	**Metrics**	**SVM**	**R Forest**	**NN-30 epochs**	**NN-100 epochs**	**CNN**
50	Precision	0.95	1.00	0.99	0.99	0.97
	Recall	1.00	0.99	1.00	1.00	0.92
	F1	0.97	1.00	1.00	0.99	0.95
	Accuracy	98.2%	99.70%	99.68%	99.54%	96.53%
100	Precision	0.99	1.00	0.99	0.99	0.99
	Recall	1.00	1.00	1.00	1.00	0.97
	F1	0.99	1.00	1.00	1.00	0.98
	Accuracy	99.56%	99.86%	99.74%	99.72%	98.56%
200	Precision	0.99	1.00	0.99	0.99	0.99
	Recall	1.00	0.99	1.00	1.00	0.98
	F1	1.00	1.00	1.00	1.00	0.98
	Accuracy	99.74%	99.7%	99.79%	99.79%	98.98%
300	Precision	0.99	1.00	0.99	0.99	0.98
	Recall	1.00	0.99	1.00	1.00	0.98
	F1	1.00	1.00	1.00	1.00	0.98
	Accuracy	99.71%	99.79%	99.76%	99.77%	98.82%

Table 4: Classification results for Google dataset for GloVe dimensions 50, 100, 200, 300 with SVM polynomial kernel (3 degrees of freedom), Random Forest-100 trees, NN 30 and 100 epochs, and CNN 10 epochs.

Ultimately, the results from Tables 4, 5 and 6 definitely outperform all the baselines involved in our comparison and show that:

1. Learning how to classify analogy leads to much better results than modifying or constraining the parallelogram formula.
2. Dimension 100 for GloVe word embedding is also more than enough to outperform other approaches, whatever the dataset.

Class	**Metrics**	**SVM**	**R Forest**	**NN-30**	**NN-100**	**CNN**
E1-10	Precision	0.94	0.98	0.97	0.97	0.89
	Recall	1.00	0.96	0.97	0.97	0.87
	F1	0.97	0.97	0.97	0.97	0.87
	Accuracy	97.90%	97.93%	97.91%	97.92%	91.61%
L1-10	Precision	0.83	0.96	0.95	0.97	0.86
	Recall	1.00	0.96	0.94	0.97	0.85
	F1	0.90	0.96	0.94	0.97	0.85
	Accuracy	90.27%	97.22%	96.28%	97.96%	90.02%
I1-10	Precision	0.98	1.00	0.99	0.99	0.98
	Recall	1.00	1.00	1.00	1.00	0.94
	F1	0.99	1.00	1.00	1.00	0.96
	Accuracy	99.46%	99.74%	99.53%	99.50%	97.34%
D1-10	Precision	0.86	1.00	1.00	0.99	0.92
	Recall	1.00	1.00	1.00	1.00	0.94
	F1	0.91	1.00	1.00	1.00	0.93
	Accuracy	92.14%	100.00%	99.86%	99.84%	95.2%

Table 5: Classification results for BATS dataset for GloVe dimension 100 with SVM polynomial kernel (3 degrees of freedom), Random Forest-100 trees, NN 30 and 100 epochs, and CNN (dimension 50) 5 epochs.

5.2. Analogy completion as a regression task

Here, we also used $\mathbb{R}^n$ word embeddings with $n \in \{50, 100, 200, 300\}$. The network proposed in Section 3.4 was trained for 50 epochs 10 fold cross-validation. As suggested in Subsection 3.1, the results for NN have to be compared with the works of [19] for $3CosMul$ (code provided by [30]) and $LRCos$ (code provided by [15]).

The results are in Table 7 for Google dataset, Table 8 for the BATS dataset and Table 9 for the DiffVec dataset. As seen from all three tables, NN regression outperforms $3CosMul$ and $LRCos$ by a large margin - $LRCos$ was implemented using leave-one-out cross validation. However, $3CosMul$ and $LRCos$ have some advantages over NN: they are much faster to execute and require less resources. Note that the NN was trained for each category. In our previous work [31], the NN model was trained on all categories for the Google dataset, but we feel that it would be a fairer comparison with $LRCos$ if the NN was trained for each category, as the relation in $LRCos$ was extracted for each category.

In Section 3.4, we explained that both NN and $LRCos$ try to exploit the relationship of d with a, b and c (as $3CosMul$). In $LRCos$, a (e.g., 'Japan') serves as

Class	Metrics	SVM	R Forest	NN-30	NN-100	CNN
Object:State	P	0.62	1.00	0.99	0.99	0.85
(Class 2)	R	1.00	1.00	1.00	1.00	0.75
	F1	0.76	1.00	0.99	0.99	0.79
	Acc	79.39%	99.91%	99.56%	99.60%	87.12%
Compensatory	P	0.61	0.86	0.85	0.84	0.75
Action	R	0.97	0.85	0.86	0.86	0.84
(Class 6)	F1	0.75	0.86	0.85	0.85	0.79
	Acc	78.04%	90.62%	90.2%	90.05%	85.00%
Intended	P	0.59	1.00	0.98	0.97	0.95
Action	R	1.00	0.99	1.00	1.00	0.90
(Class 10)	F1	0.74	0.99	0.99	0.99	0.92
	Acc	76.49%	99.63%	99.19%	99.08%	94.87%
Collective	P	0.99	1.00	0.99	0.99	0.76
Noun	R	1.00	0.99	1.00	0.99	0.88
(Class 12)	F1	0.99	0.99	0.99	0.99	0.81
	Acc	99.56%	99.65%	99.59%	99.59%	86.49%
Average	P	0.68	0.95	0.94	0.94	0.78
SVM 32 classes	R	1.00	0.92	0.95	0.96	0.73
others 34 classes	F1	0.80	0.93	0.94	0.94	0.75
	Acc	79.70%	96.48%	95.89%	96.07%	83.56%

Table 6: Classification results for DiffVec dataset for GloVe dimension 100 with SVM polynomial kernel (3 degrees of freedom), Random Forest-100 trees, NN 30 and 100 epochs, and CNN (dimension 50) 5 epochs. Full results can be found in Appendix C.

a negative example and b (e.g., 'Tokyo') serves as a positive example, and that d should be close to c (e.g., 'France'), but the relationships perhaps may not be well captured during the regression process. Using NN, the relationship of d with a, b and c is captured under the form $g(f_1(a,b), f_2(a,c))$ via the NN structure. This may explain the better results obtained with NN.

Tables 7, 8 and 9 also demonstrate that the accuracies of both $3CosMul$ and $LRCos$ increase as the dimension of the embedding increases. Interestingly, this is not the case for NN: its accuracies are less dependent on the dimension - only in the DiffVec dataset that the average accuracy increases by 3.85% from dimension 50 to dimension 100, and the increase is insignificant (only 0.02%) from dimension 100 to dimension 200. This appears to support the argument put forward by Drodz et al. [15] that a better method can overcome the shortcoming of an embedding. Up to a point, this argument is acceptable, when the relationships are simple, such as those presented in the Google, BATS and DiffVec datasets. Nevertheless, when the relationships are very complex, such as those in SAT, then it would be difficult to capture those relationships using NN as discussed in Section 6.

	NN Regression				$3CosMul$				$LRCos$			
Dim	50	100	200	300	50	100	200	300	50	100	200	300
Average	97.31%	**97.49%**	97.49%	97.48%	34.89%	56.64%	61.84%	64.81	46.55%	63.85%	69.12%	70.24%
Com capitals	100.00%	100.00%	100.00%	100.00%	63.8%	80.0%	86.9%	88.9%	91.30%	95.65%	95.65%	95.60%
All capitals	99.75%	100.00%	100.00%	100.00%	50.0%	77.0%	87.5%	90.1%	85.34%	95.69%	99.14%	98.28%
US cities	67.56%	**69.37%**	69.10%	69.16%	6.9%	17.0%	29.8%	36.5%	26.92%	38.46%	62.82%	62.82%
Currencies	94.97%	95.47%	**95.69%**	95.58%	5.0%	15.0%	22.5%	24.4%	16.67%	23.33%	26.67%	26.67%
Nationalities	100.00%	100.00%	100.00%	100.00%	84.7%	89.1%	94.1%	94.6%	85.37%	90.24%	92.68%	92.68%
Gender	100.00%	100.00%	100.00%	100.00%	61.5%	79.4%	86.6%	88.6%	4.36%	5.45%	6.91%	6.55%
Plurals	100.00%	100.00%	100.00%	100.00%	41.4%	71.9%	79.8%	83.3%	67.56%	75.68%	86.49%	83.78%
Base-gerund	100.00%	100.00%	100.00%	100.00%	35.2%	65.6%	68.1%	71.0%	36.36%	81.82%	78.79%	81.82%
Gerund-past	100.00%	100.00%	100.00%	100.00%	27.3%	53.1%	59.6%	62.5%	37.50%	75.00%	82.50%	85.00 %
Base to 3^{rd}	100.00%	100.00%	100.00%	100.00%	28.8%	59.1%	64.9%	68.4%	43.33%	66.67%	80.00%	80.00%
Adj-adverb	100.00%	100.00%	100.00%	100.00%	10.8%	22.0%	22.9%	23.4%	31.25%	50.00%	50.00%	50.00%
Adj-Comprtv	100.00%	100.00%	100.00%	100.00%	48.2%	68.5%	74.1%	76.5%	59.46%	81.00%	89.19%	89.19%
Adj-Superltv	100.00%	100.00%	100.00%	100.00%	18.6%	50.1%	67.5%	73.7%	55.88%	73.53%	85.29%	85.29%
Adj un-prefix	100.00%	100.00%	100.00%	100.00%	6.2%	17.1%	21.5%	25.4%	10.34%	41.38%	44.83%	44.83%

Table 7: Accuracy of regression (analogy completion task) using NN, $3CosMul$ and $LRCos$ for Google dataset. The best results are in bold (unless the accuracy is 100%).

	NN Regression				$3CosMul$				$LRCos$			
Dim	50	100	200	300	50	100	200	300	50	100	200	300
E1-10	74.71%	75.74%	75.94%	**75.95%**	8.68%	14.64%	18.54%	20.35%	24.69%	29.30%	37.72%	37.31%
L1-10	90.11%	90.77%	90.82%	**90.68%**	1.70%	4.47%	5.9%	6.3%	5.63%	10.86%	11.67%	13.28%
I1-10	99.99%	100.00%	100.00%	100.00%	32.41%	55.31%	60.20%	63.11%	48.09%	69.69%	73.53%	73.73%
D1-10	100.00%	100.00%	100.00%	100.00%	3.25%	8.37%	10.47%	11.92%	15.96%	26.76%	27.59%	27.40%

Table 8: Accuracy of regression (analogy completion task) using NN, $3CosMul$ and $LRCos$ for the BATS dataset. This dataset has four categories: E-Encyclopedia, L-Lexicography, I-Inflectional Morphology, D-Derivational Morphology. Each category has ten subcategories. The table above shows the average result of each category. The best results are in bold (unless the accuracy is 100%).

	NN Regression				$3CosMul$				$LRCos$			
Dim	50	100	200	300	50	100	200	300	50	100	200	300
Object:State (Class 2)	97.68%	**98.08**%	97.45%	97.22%	0.40%	0.52%	0.29%	0.34%	0.00%	6.67%	3.33%	3.33%
Compensatory Action (Class 6)	87.70%	**88.50**%	88.19%	88.12%	0.00%	0.20%	0.26%	0.26%	0.00%	0.00%	3.57%	3.57%
Intended Action (Class 10)	**93.62%**	92.37%	91.64%	91.51%	0.39%	2.50%	3.82%	5.00%	0.00%	15.00%	25.00%	15.00%
Collective Noun (Class 12)	72.48%	79.99%	**80.13%**	80.00%	0.03%	0.26%	0.39%	0.56%	3.97%	6.35%	6.35%	6.35%
Average	89.73%	93.58%	**93.60%**	93.40%	4.89%	8.80%	9.69%	10.21%	8.91%	16.22%	17.63%	17.84%

Table 9: The accuracy of regression (analogy completion task) using NN, $3CosMul$ and $LRCos$ for the DiffVec dataset for four sample classes and the average of 34 classes for NN and $3CosMul$ and 32 classes for $LRCos$. The best results are in bold.

6. Semantic analogies : SAT dataset

As we have previously seen, the machine learning approaches are quite successful in recognizing word analogies from Google, BATS and DiffVec datasets. Nevertheless, all the datasets are mainly made of *morpho-syntactic* analogies, not really *semantic* analogies. We are not in a position to provide a clear and concise definition of *semantic* analogies but everybody understands the difference between 'man:men::woman:women' (*morpho-syntactic*) and 'ostrich:bird::lion:cat' (*semantic*, extracted from SAT). SAT is generally considered as a well-accepted set of *semantic* analogies: this is the set of analogies used in the SAT college entrance test. It is a set of 374 multiple-choice questions where a pair of words $(a : b)$ has five options $(c : d)$ but only one option makes a valid analogy with the first pair. So we have 374 × 5 examples among which only 374 of the examples are valid *semantic* analogies. Thus, SAT provides both positive and negative examples - which is not the case for Google and BATS datasets where we only have positive examples, at least initially. Below is an example from the SAT dataset:

1. $remuneration : labor :: trophy : victory$ (the valid analogy)
2. $remuneration : labor :: gratuity : bonus$ (the 4 other candidates)
3. $remuneration : labor :: apology : regret$
4. $remuneration : labor :: pledge : donation$
5. $remuneration : labor :: debt : loan$

It appears that only 367 examples have a proper GloVe embedding: so we focus on this subset.

With his *SuperSim* (for 'supervised similarity') approach, Turney [56] achieves a score of 54.8% of correct answers for the classification task, which reveals the inherent difficulty of this dataset. In order to work with a balanced dataset, Turney selectively increases the number of positive examples by applying symmetry postulate and internal reversal property (see Section 2)[13]. Apart from classifying analogies, *SuperSim* can also perform other tasks.

6.1. Classification on SAT

For classification, we have chosen the following option: we consider only the 367 valid analogies, and we apply our dataset extension getting 367×8 positive examples and 367×16 negative examples (obtained by permutation). Then we use

[13]Applying central permutation would lead to too many positive examples.

Method 2 and 10-fold cross-validation to train and test as usual. In that context, we have a dataset of total size $367 \times 24 = 8,808$ (positive and negative) examples.

As we have a ratio of 1 valid analogy for 2 non-valid ones, a baseline algorithm classifying all quadruples $a : b :: c : d$ as non-valid would have 66% accuracy. The results are in Table 10. We observe that the best performer is CNN with an accuracy larger than 77% for 10 epochs whatever the GloVe dimension. This could indicate that CNN is able to capture more complex relationships between words than a simple NN. This is obviously the case when dealing with images instead of words!

Obviously, as expected, we are less accurate than with Google, BATS and DiffVec datasets. Nevertheless, we are better than what has been done in [56], where accuracy is in the range of 55%.

	Dim	**Metrics**	**SVM**	**R Forest**	**NN-30**	**NN-100**	**CNN**
SAT $\#TS = 8,808$	50	Precision	0.37	0.25	0.45	0.45	0.69
		Recall	0.99	0.00	0.45	0.37	0.64
		F1	0.54	0.00	0.45	0.40	0.66
		Accuracy	44.21%	66.53%	63.16%	63.93%	78.33%
	100	Precision	0.39	0.55	0.44	0.44	0.69
		Recall	0.98	0.01	0.46	0.44	0.60
		F1	0.55	0.01	0.45	0.44	0.62
		Accuracy	46.94%	66.82%	62.57%	62.90%	77.63%
	200	Precision	0.62	0.44	0.44	0.45	0.77
		Recall	0.38	0.00	0.50	0.48	0.51
		F1	0.47	0.01	0.47	0.46	0.58
		Accuracy	71.39%	66.67%	62.05%	63.05%	77.62%
	300	Precision	0.41	0.49	0.45	0.45	0.75
		Recall	0.04	0.00	0.50	0.47	0.59
		F1	0.06	0.01	0.48	0.46	0.62
		Accuracy	67.32%	66.67%	63.17%	63.36%	78.61%

Table 10: Classification results for SAT dataset using SVM polynomial kernel (3 degrees of freedom), Random Forest-100 trees, NN 30 and 100 epochs and CNN 10 epochs.

6.2. Completion on SAT

The completion task on SAT was conducted on the 367 examples including only the 367 valid analogies, without taking into account the negative examples given by SAT. We run the three analogy completion algorithms ($3CosMul$,

$LRCos$[14] and NN), and they all perform poorly.

This leads us to concentrate on a more modest task where we take advantage of the negative examples provided in SAT. We proceed as follows. When we have a complete example $a : b : c_1 : d_1$ (valid analogy) and four incorrect analogies,

$$a : b : c_2 : d_2, a : b : c_3 : d_3, a : b : c_4 : d_4, a : b : c_5 : d_5,$$

Then, we compute the five cosine similarities of d with $d_1, ..., d_5$ so, it yields

$$sim(d, d_1), sim(d, d_2), sim(d, d_3), sim(d, d_4), sim(d, d_5).$$

Note that we have preferred to use d_2, ..., d_5 from SAT instead of randomly chosen words in the vocabulary. We used the three methods to find the solution d. If $sim(d, d_1)$ is the largest one, this is a success for the method, a failure otherwise. Then the accuracy is computed with these notions of success/failure. This allows us to compare the three methods $3CosMul$, $LRCos$ and NN on this new basis. Results are given in Table 11.

	Dim	$3CosMul$	$LRCos$	NN
SAT	50	53.51%	48.92%	35.78%
	100	59.73%	54.32%	33.99%
	200	63.85%	60.00%	32.64%
	300	65.27%	59.73%	36.22%

Table 11: Accuracy of regression (analogy completion task) using $3CosMul$, $LRCos$ and NN (50 epochs) on SAT. The accuracies for $3CosMul$ and $LRCos$ are comparable, and the accuracies for NN are the lowest.

The results for $3CosMul$ and $LRCos$ are comparable, with $3CosMul$ always slightly better than $LRCos$. The machine learning approach NN is no longer successful here. This might be due to the very limited number of examples: machine learning methods such as NN usually need a large amount of data to be successful.

7. Related work

First of all, there are slightly different views of what is an analogical proportion between words. Following the postulates recalled in Section 2, it is legitimate to expect that central permutation preserves the quaternary analogy relation. When word analogy is viewed as a byproduct of binary relation induction,

[14]Mind that $LRCos$ uses negative examples but they are not the SAT negative examples, but rather words that are different from the target class.

it is clear that central permutation is not a concern. However, the parallelogram paradigm implicitly assumes central permutation ($a - b = c - d$) is equivalent to $a - c = b - d$). Nevertheless, in the context of word analogy, central permutation does not always lead to a cognitively acceptable analogical proportion. This is especially the case when a, c on the one hand and b, d on the other hand different categories, as already pointed out in subsection 2.4.

In the computational linguistics literature, the word analogy trend assumes that the prediction of a new word pair in relation induction always leads to a word analogy with any other pair sharing the same relation. Following this view, works such that [39, 20, 49] extract word pairs representative of semantic relations from a large corpora, which then serve as input to learning models in order to predict new instances of the considered relation using the given corpus. In the same spirit, we can also cite [61, 60] for the recognition of lexical semantic relations. Another option is explored in [23] where relation vectors that encode the relationship between two words are directly embedded and learnt from co-occurrence statistics (already used in [1]); the approach is experimented on relational induction among other tasks. However, these authors do not primarily focus on analogy even if classification and/or completion of analogy can be used to test some models.

Besides, we have to stress the fact that, apart from Turney's works [55], we are unaware of recent works completely departing from the vector offset methods to detect analogy. As we have seen in subsection 3.1:

- The analogy models from [38, 30, 19] were developed by tuning the standard additive formula $a - b = c - d$, leading to a much more successful formula ($3CosMul$ in Equation 1).
- The analogy models from [15, 18] or the translation model of [8] introduce a constraint on the candidate solutions. It appears that the translation model outperforms all other methods they compared with.
- Finally, the regression model from [8] slightly relaxes the assumption but still assuming some kind of simple underlying relation. It can outperform the translation model on some specific categories.

Nevertheless, the problem of recognizing word analogies has been attempted well before the emergence of effective word embedding systems: we can cite [48, 53, 58, 55, 54] for instance. The datasets were generally limited in terms of size with regard to the current standard. For instance [55] uses SAT corpus: the highest accuracy on this set is 56.1% knowing that a random guess yields an accuracy of 20% and senior high school students get 57%. These results are quite interesting as the dataset mainly consists of semantic analogies like *mason:stone::carpenter:wood*. More recently, researchers were more interested in working on the analogy completion problem, as with word embedding techniques, numerical tools became widely available. Instead of dealing with discrete space of words, they deal with continuous space language models.

From another perspective, the BART (standing for Bayesian Analogy with Relational Transformations) model described in [34], is also based on neural-network but more dedicated to relation induction i.e. finding pairs (c, d) when a pair (a, b) is given.

From another viewpoint, some researchers have focused on the task of transforming a query image according to an example pair of related images [47]. They have taken their inspiration from language modeling techniques. They focus on generating an appropriate image to make a valid analogy between images. Their approach is a two-step learning process not far from what we did:

- firstly, learn a deep encoder function f from $\mathbb{R}^d$ to $\mathbb{R}^k$ that maps images to an embedding space suitable for reasoning about analogies. In some sense, it is the counterpart of word2vec or GloVe for images. In the target space, they solve the analogical equation using $f(b) - f(a) + f(c)$

- secondly, learn a deep decoder function g from $\mathbb{R}^k$ to $\mathbb{R}^d$ to ultimately solve the equation $a : b :: c : x$ with $g(f(b) - f(a) + f(c))$.

The authors then apply their model to diverse image-related tasks, and finally to the task of analogy making on 3D car models. But, they also depart from the parallelogram viewpoint by learning from the training set a more sophisticated relation linking a, b, c to the solution d. It appears that the model can perform 3D pose transfer and rotation only by analogy. Apart from that, comparing mean-squared prediction error, their final model performs much better than the additive option, even when this additive option is modified according to the multiplicative option of [30]. We could understand this as a confirmation that it is more effective to learn the relation between (a, b, c) and the solution d than to rely on a parallelogram view.

BERT (for Bidirectional Encoder Representations from Transformers) developed by Google [14] is a recent option as a language representation model. It seems that BERT model can be fine-tuned for a wide range of tasks without huge architectural modifications. At this stage, we have not yet investigated the use of

BERT as a tool to classify or to solve analogical equations, but obviously it is a track worth of interest, and we can take inspiration from [24, 7].

8. Conclusion

In this paper, we investigate the logical axioms of analogical proportions to derive new properties rarely used in practice. This modeling highlights the fact that analogy is as much a matter of dissimilarity as a matter of similarity. When we refer to numbers or vectors instead of Boolean values, an analogy is often viewed as $a - b = c - d$ (arithmetic analogy) and may be unable to capture the complexity of similarity/dissimilarity. To try to bridge the gap between the Boolean view and the numerical one, we focus on natural language analogies using word embedding techniques. This leads us to suggest radically new approaches not only for classifying whether a quadruple of words is an analogy but also for solving analogical equations.

Previous attempts rely on predefined formulas: this can be restrictive as it assumes that we have a clear understanding of what an analogy is in natural language. In fact, we have no such understanding. Our methodology is quite different: it takes advantage of the theoretical analysis of the structural properties of analogical proportions for augmenting the training set and of recent word embedding systems in order to learn the unknown relationship amongst the words in an analogy. Embedding the words using GloVe into a multi-dimensional vector space $\mathbb{R}^n$, the matrix constituted of 4 vectors can be viewed as an image of dimension $n \times 4$. From a machine learning perspective, the permutation properties of analogical proportion lead to an increase of the dataset size by 8 for positive examples and by 16 for negative examples. Using analogies from Google, BATS and DiffVec datasets, we have implemented several machine learning classifiers (SVM, Random Forest, NN and CNN) with an average (on all categories) accuracy higher than 96% for Google, higher than 90% for BATS and higher than 79% for DiffVec whatever the tested machine learning methods (note that if we ignore SVM and CNN on DiffVec, the average accuracy is higher that 95%). However, the accuracies on the SAT dataset were lower, but it is still up to 77% for the most successful method, namely CNN. SAT is indeed much more "semantic" than Google, BATS and DiffVec, which probably makes it more challenging.

Regarding the analogy completion task, we also depart from the classical views derived from the arithmetic analogy using a formula. Instead, we try to learn such a definition via machine learning techniques.

Experiment results show that the machine learning methods outperform the formula-inspired methods for classification task (all four datasets) and analogy completion task (three out of four datasets). Machine learning methods might be useful for solving more semantic analogies.

Due to the success of the machine learning approaches with GloVe embedding, one could ask what we could achieve with a complete end-to-end learning approach, where the initial embedding is built via a neural network. This is an open question at this stage and in the future, it is worth to conduct a deeper investigation.

Another line of future investigation would be to couple maybe the approaches proposed with works trying to recognize the role of entities or the conceptual space they belong to (e.g., [6]) in the analogical proportions. This might be of particular interest for dealing with 'semantic' proportions.

Still we have to acknowledge the fact that while machine learning techniques may be very powerful tools for dealing with analogy classification and analogy completion tasks, this does not provide any contribution on what is exactly an analogy between words. It is likely that a strict judgement for deciding if an analogical proportion is valid or not, or if is a matter of degree (or of context), would require more high level information than the one provided by embeddings.

Acknowledgements The authors are deeply indebted to the two reviewers who provided many useful suggestions and references. The paper has substantially benefited from their feedback. Thanks are also due to Peter Turney for kindly providing the SAT dataset.

References

[1] S. Arora, Y. Li, Y. Liang, T. Ma, and A. Risteski. A latent variable model approach to PMI-based word embeddings. Trans. Assoc. Comput. Linguistics, 4:385–399, 2016.

[2] N. Barbot, L. Miclet, and H. Prade. Analogy between concepts. Artif. Intell., 275:487–539, 2019.

[3] M. Bayoudh, H. Prade, and G. Richard. Evaluation of analogical proportions through Kolmogorov complexity. Knowledge-Based Systems, 29:20–30, 2012.

[4] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko. Translating embeddings for modeling multi-relational data. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, Advances in Neural Information Processing Systems 26, pages 2787–2795. Curran Associates, Inc., 2013.

[5] M. Bounhas, H. Prade, and G. Richard. Analogy-based classifiers for nominal or numerical data. Int. J. Approx. Reasoning, 91:36–55, 2017.

[6] Z. Bouraoui, J. Camacho-Collados, L. Espinosa-Anke, and S. Schockaert. Modelling semantic categories using conceptual neighborhood. In Proc. 34th AAAI Conf. on Artificial Intelligence, AAAI'20, New York, Feb. 7-12, 2020, pages 7448–7455. AAAI Press, 2020.

[7] Z. Bouraoui, J. Camacho-Collados, and S. Schockaert. Inducing relational knowledge from BERT. In Proc. 34th AAAI Conf. on Artificial Intelligence (AAAI' 20, New York, February 7-12, pages 7456–7463. AAAI Press, 2020.

[8] Z. Bouraoui, S. Jameel, and S. Schockaert. Relation induction in word embeddings revisited. In E. M. Bender, L. Derczynski, and P. Isabelle, editors, Proc. 27th Int. Conf. on Computational Linguistics, COLING'18, Santa Fe, New Mexico, Aug. 20-26, 2018, pages 1627–1637. Association for Computational Linguistics, 2018.

[9] L. Breiman. Random forests. Mach. Learn., 45(1):5–32, 2001.

[10] C.Allen and T. M. Hospedales. Analogies explained: Towards understanding word embeddings. In K. Chaudhuri and R. Salakhutdinov, editors, Proc. 36th Int. Conf. on Machine Learning, ICML'19, 9-15 June , Long Beach, Ca., volume 97, pages 223–231. PMLR, 2019.

[11] D. Chen, J. C. Peterson, and T. L. Griffiths. Evaluating vector-space models of analogy. CoRR, abs/1705.04416, 2017.

[12] W. Correa Beltran, H. Prade, and G. Richard. Constructive solving of Raven's IQ tests with analogical proportions. Int. J. Intell. Syst., 31(11):1072–1103, 2016.

[13] C. Cortes and V. Vapnik. Support-vector networks. Machine Learning, 20(3):273–297, 1995.

[14] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. CoRR, abs/1810.04805, 2018.

[15] A. Drozd, A. Gladkova, and S. Matsuoka. Word embeddings, analogies, and machine learning: Beyond king - man + woman = queen. In Proc. COLING'16, 26th Int. Conf. on Comput. Linguistics: Tech. Papers, pages 3519–3530. The COLING 2016 Organizing Committee, 2016.

[16] D. Dubois, H. Prade, and G. Richard. Multiple-valued extensions of analogical proportions. Fuzzy Sets and Systems, 292:193–202, 2016.

[17] D. Gentner. The mechanisms of analogical learning. In S. Vosniadou and A. Ortony, editors, Similarity and Analogical Reasoning, pages 197–241. Cambridge University Press, New York, 1989.

[18] A. Gladkova, A. Drozd, and S. Matsuoka. Analogy-based detection of morphological and semantic relations with word embeddings: What works and what doesn't. In Proc. NAACL-HLT SRW, pages 47–54, San Diego, Ca., June 12-17, 2016, 2016. ACL.

[19] Y. Goldberg and O. Levy. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. CoRR, abs/1402.3722, 2014.

[20] I. Hendrickx, S. N. Kim, Z. Kozareva, P. Nakov, D. Ó Séaghdha, S. Padó, M. Pennacchiotti, L. Romano, and S. Szpakowicz. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In K. Erk and C. Strapparava, editors, Proc. 5th Int. Workshop on Semantic Evaluation, SemEval@ACL 2010, Uppsala, July 15-16, 2010, pages 33–38. The Association for Computer Linguistics, 2010.

[21] M. Hesse. On defining analogy. Proceedings of the Aristotelian Society, 60:79–100, 1959.

[22] D. Hofstadter and M. Mitchell. The Copycat project: A model of mental fluidity and analogy-making. In D. Hofstadter and The Fluid Analogies Research Group, editors, Fluid Concepts and Creative Analogies: Computer Models of the Fundamental Mechanisms of Thought, pages 205–267, New York, NY, 1995. Basic Books, Inc.

[23] S. Jameel, Z. Bouraoui, and S. Schockaert. Unsupervised learning of distributional relation vectors. In I. Gurevych and Y. Miyao, editors, Proc. 56th Annual Meeting of the Assoc. for Comput. Linguistics, ACL'18, Melbourne, July 15-20, Vol. 1: Long Papers, pages 23–33, 2018.

[24] G. Jawahar, B. Sagot, and D. Seddah. What does BERT learn about the structure of language? In A. Korhonen, D. R. Traum, and Ll. Màrquez, editors, Proc. 57th Conf. of the Assoc. for Comput. Linguistics, ACL 2019, Florence, July 28- Aug. 2, 2019, Vol. 1: Long Papers, pages 3651–3657. Association for Computational Linguistics, 2019.

[25] M. Joshi, E. Choi, O. Levy, D. S. Weld, and L. Zettlemoyer. pair2vec: Compositional word-pair embeddings for cross-sentence inference. In J. Burstein, Ch. Doran, and T. Solorio, editors, Proc. 2019 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, June 2-7, 2019, Volume 1 (Long and Short Papers), pages 3597–3608. Association for Computational Linguistics, 2019.

[26] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2014.

[27] Y. Lepage. Analogy and formal languages. Electr. Notes Theor. Comput. Sci., 53, 2001.

[28] Y. Lepage, J. Migeot, and E. Guillerm. A measure of the number of true analogies between chunks in Japanese. In Z. Vetulani et al., editor, Human Language Technology. Challenges of the Information Society, LNCS 5603, 154-164. Springer, 2009.

[29] O. Levy and Y. Goldberg. Dependency-based word embeddings. In Proc. 52nd Annual Meeting of Ass. Comp. Ling. (Vol 2: Short Papers), pages 302–308, 2014.

[30] O. Levy, Y. Goldberg, and I. Dagan. Improving distributional similarity with lessons learned from word embeddings. Trans. Ass. for Comput. Ling., 3:211–225, 2015.

[31] S. Lim, H. Prade, and G. Richard. Solving word analogies: A machine learning perspective. In G. Kern-Isberner and Z. Ognjanovic, editors, Proc. 15th Europ. Conf. on Symbolic and Quantitative Approaches to Reasoning

with Uncertainty (ECSQARU'19), Belgrade, Sept. 18-20, volume 11726 of LNCS, pages 238–250. Springer, 2019.

[32] Tal Linzen. Issues in evaluating semantic spaces using word analogies. CoRR, abs/1606.07736, 2016.

[33] A. Lovett, K. Forbus, and J. Usher. A structure-mapping model of Raven's progressive matrices. In Proc. 32nd Annual Conf. of the Cognitive Science Soc., Portland, OR, 2010.

[34] H. Lu, Y Wu, and K.H. Holyoak. Emergence of analogy from relation learning. In Proc. of the National Academy of Sciences, volume 116, pages 4176–4181, 2019.

[35] L. Miclet, S. Bayoudh, and A. Delhay. Analogical dissimilarity: Definition, algorithms and two experiments in machine learning. JAIR, 32:793–824, 2008.

[36] T. Mikolov, K. Chen, G. S Corrado, and J. Dean. Efficient estimation of word representations in vector space. CoRR, abs/1301.3781, 2013.

[37] T. Mikolov, I. Sutskever, K. Chen, G. S Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges et al., editor, Advances in Neural Information Processing Systems 26, pages 3111–3119. Curran Associates Inc., 2013.

[38] T. Mikolov, W. Yih, and G. Zweig. Linguistic regularities in continuous space word representations. In Proc. Conf. of North Amer. Chap. of Ass. for Comp. Ling.: Human Language Technologies, pages 746–751, 2013.

[39] M. Mintz, S. Bills, R. Snow, and D. Jurafsky. Distant supervision for relation extraction without labeled data. In K.-Y. Su, J. Su, and J. Wiebe, editors, ACL 2009, Proc. of the 47th Annual Meeting of the Assoc. for Comput. Linguistics and the 4th Int. Joint Conf. on Natural Language Processing of the AFNLP, 2-7 Aug., Singapore, pages 1003–1011. The Association for Computer Linguistics, 2009.

[40] M. A. Nielsen. Neural Networks and Deep Learning. Determination Press, 2018.

[41] M. Nissim, R. van Noord, and R. van der Goot. Fair is better than sensational: Man is to doctor as woman is to doctor. Comput. Linguistics, 46(2):487–497, 2020.

[42] A. Paccanaro and G. E. Hinton. Learning distributed representations of concepts using linear relational embedding. IEEE Trans. on Knowl. and Data Eng., 13(2):232–244, March 2001.

[43] J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In Proc. Conf. on Empirical Methods in Natural Language Processing (EMNLP), volume 14, pages 1532–1543. Association for Computational Linguistics, 01 2014.

[44] J. Pennington, R. Socher, and Ch. D. Manning. Glove: Global vectors for word representation. In Empirical Methods in Natural Language Processing (EMNLP), pages 1532–1543, 2014.

[45] H. Prade and G. Richard. From analogical proportion to logical proportions. Logica Univers., 7:441–505, 2013.

[46] N. Rao, Ch. Cox, R. Nowak, and T. Rogers. Sparse overlapping sets lasso for multitask learning and its application to fMRI analysis. In Ch. J. C. Burges, L. Bottou, Z. Ghahramani, and K. Q. Weinberger, editors, Advances in Neural Information Processing Systems 26: Proc. 27th Annual Conf. on Neural Information Processing Systems, Dec. 5-8, Lake Tahoe, Nevada, pages 2202–2210, 2013.

[47] S. E. Reed, Y. Zhang, Y.t. Zhang, and H.l. Lee. Deep visual analogy-making. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, Advances in Neural Information Processing Systems 28, pages 1252–1260. Curran Associates, Inc., 2015.

[48] W. R. Reitman. Cognition and thought. an information processing approach. Psych. in the Schools, 3(2), 1965.

[49] S. Riedel, L. Yao, and A. McCallum. Modeling relations and their mentions without labeled text. In J. L. Balcázar, F. Bonchi, A. Gionis, and M. Sebag, editors, Machine Learning and Knowledge Discovery in Databases, European Conference, ECML PKDD'10, Barcelona, Sept. 20-24, Proc., Part III, volume 6323 of LNCS, pages 148–163. Springer, 2010.

[50] Richard Robinson. Plato's consciousness of fallacy. Mind, 51(202):97–114, 1942.

[51] D. E. Rumelhart and A. A. Abrahamson. A model for analogical reasoning. Cognitive Psychol., 5:1–28, 1973.

[52] R. Tibshirani. The lasso method for variable selection in the Cox model. Statistics in Medicine, 16(4):385–395, 1997.

[53] P. D. Turney. Similarity of semantic relations. Computational Linguistics, 32(3):379–416, 2006.

[54] P. D. Turney. The latent relation mapping engine: algorithm and experiments. JAIR, 33:615–655, 2008.

[55] P. D. Turney. A uniform approach to analogies, synonyms, antonyms, and associations. In Proc. 22nd Int. Conf. on Computational Linguistics - Vol. 1 (COLING '08), pages 905–912. Association for Computational Linguistics, 2008.

[56] P. D. Turney. Distributional semantics beyond words: Supervised learning of analogy and paraphrase. Transactions of the Association for Computational Linguistics, 1:353–366, 2013.

[57] P. D. Turney, M. L. Littman, J. Bigham, and V. Shnayder. Combining independent modules to solve multiple-choice synonym and analogy problems. CoRR, cs.CL/0309035, 2003.

[58] T. Veale. Re-representation and creative analogy: A lexico-semantic perspective. New Generation Computing, 24(3):223–240, 2006.

[59] E. Vylomova, L. Rimell, T. Cohn, and T. Baldwin. Take and took, gaggle and goose, book and read: Evaluating the utility of vector differences for lexical relation learning. In Proc. 54th Annual Meeting of the Assoc. for Comput. Linguistics (Vol. 1: Long Papers), pages 1671–1682, Berlin, 2016. Association for Computational Linguistics.

[60] K. Washio and T. Kato. Filling missing paths: Modeling co-occurrences of word pairs and dependency paths for recognizing lexical semantic relations. In M. A. Walker, H. Ji, and A. Stent, editors, Proc. 2018 Conf. of the North American Chapter of the Association for Computational Linguistics: Human

Language Technologies, NAACL-HLT 2018, New Orleans, June 1-6, 2018, Vol. 1 (Long Papers), pages 1123–1133. Association for Computational Linguistics, 2018.

[61] K. Washio and T. Kato. Neural latent relational analysis to capture lexical semantic relations in a vector space. In E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, editors, Proc. 2018 Conf. on Empirical Methods in Natural Language Processing, Brussels, Oct. 31 - Nov. 4, pages 594–600. Association for Computational Linguistics, 2018.

Appendix A. Detailed results on Google dataset for classification

		Average accuracy in %			
Class name	**#epochs**	**50**	**100**	**200**	**300**
Common capitals (1)	3	88.32%	91.62%	87.65%	85.97%
	5	94.66%	97.41%	95.28%	93.93%
	10	95.69%	98.41%	99.58%	93.93%
All capitals (2)	3	99.86%	99.97%	99.98%	99.98%
	5	99.91%	99.97%	99.98%	99.99%
	10	99.92%	99.99%	99.99%	100.0%
Currencies (4)	3	93.94%	98.06%	99.41%	99.83%
	5	95.45%	98.57%	99.86%	99.80%
	10	96.93%	99.43%	99.93%	99.97%
US cities (3)	3	83.56%	83.60%	80.69%	87.38%
	5	88.29%	91.17%	93.08%	93.47%
	10	94.61%	97.17%	99.00 %	99.20%
Gender (11)	3	98.40%	99.07%	99.69%	99.44%
	5	99.78%	99.94%	100.0%	99.99%
	10	99.83%	99.93%	99.96%	99.97%
Adj to adverb (5)	3	80.42%	82.26%	82.41%	85.49%
	5	85.42%	86.21%	91.11%	90.64%
	10	94.13%	94.92%	97.84%	97.80%
Opposite (13)	3	89.18%	84.80%	78.93%	80.99%
	5	96.50%	95.23%	90.39%	91.45%
	10	99.33%	99.60%	97.44%	98.54%
Comparative(10)	3	79.74%	83.55%	81.20%	78.01%
	5	80.32%	92.17%	90.84%	91.33%
	10	91.16%	98.34%	98.34%	97.94%
Superlative (12)	3	79.34%	83.96%	87.12%	88.48%
	5	84.34%	96.02%	97.48%	98.02%
	10	98.88%	98.88%	98.97%	99.52%
Base to gerund (14)	3	87.04%	90.46%	89.32%	89.08%
	5	91.22%	91.36%	91.48%	96.29%
	10	95.68%	97.45%	98.02%	99.21%
Nationalities (6)	3	85.00%	82.45%	78.64%	82.68%
	5	92.20%	90.84%	90.23%	87.31%
	10	96.11%	98.48%	97.30%	98.35%
Gerund to past (8)	3	91.02%	96.21%	97.56%	97.76%
	5	98.16%	98.52%	99.61%	99.73%
	10	99.35%	99.75%	99.78%	99.81%
Plurals (9)	3	96.94%	98.37%	98.01%	99.16%
	5	98.64%	99.49%	99.62%	99.67%
	10	99.54%	99.55%	99.71%	99.72%
Base to 3^{rd} person (7)	3	80.49%	79.27%	81.6%	77.63%
	5	87.31%	87.54%	85.21%	89.51%
	10	95.05%	97.65%	97.45%	98.45%
Full Google (15)	3	96.01%	97.79%	97.85%	97.93%
	5	96.01%	98.07%	98.65%	98.84%
	10	96.49%	98.45%	98.96%	98.88%

Table A.12: Classification results for CNN on Google categories per GloVe dimension.

Appendix B. Detailed results on BATS dataset for classification

Taking into account our experiments with Google dataset, we have experimented per BATS subcategory (40 subcategories in total) with the following parameters:

- CNN classifier only (as CNN is the most successful on Google)
- Glove dimension 50 (as it is enough for the CNN to get good results on Google)
- Number of epochs 5 (as it is still enough for the CNN to get good results on Google).

As we know, the most challenging category is L, with subcategory L10 very tough to capture. But the CNN provides acceptable results. On subcategory L10, CNN still leads to an accuracy close to 70%.

Class name	Metrics	Values
I01 [noun - plural-reg]	Precision	0.99
	Recall	0.93
	F1	0.96
	Accuracy	97.39% (+/- 0.75%)
I02 [noun - plural-irreg]	Precision	0.90
	Recall	0.85
	F1	0.88
	Accuracy	91.97% (+/- 1.46%)
I03 [adj - comparative]	Precision	0.99
	Recall	0.99
	F1	0.99
	Accuracy	99.26% (+/- 0.41%)
I04 [adj - superlative]	Precision	0.99
	Recall	1.00
	F1	0.99
	Accuracy	99.53% (+/- 0.25%)
I05 [verb-inf - 3pSg]	Precision	1.00
	Recall	1.00
	F1	1.00
	Accuracy	99.88% (+/- 0.17%)
I06 [verb-inf - Ving]	Precision	1.00
	Recall	0.90
	F1	0.95
	Accuracy	96.80% (+/- 2.34%)
I07 [verb-inf - Ved]	Precision	1.00
	Recall	0.99
	F1	0.99
	Accuracy	99.60% (+/- 0.22%)
I08 [verb-Ving - 3pSg]	Precision	0.97
	Recall	0.92
	F1	0.94
	Accuracy	96.41% (+/- 0.84%)
I09 [verb-Ving - Ved]	Precision	0.96
	Recall	0.84
	F1	0.90
	Accuracy	93.65% (+/- 2.14%)
I10 [verb-3pSg - Ved]	Precision	1.00
	Recall	0.97
	F1	0.98
	Accuracy	98.93% (+/- 0.40%)

Class name	Metrics	Values
D01 [noun+less-reg]	Precision	0.82
	Recall	0.94
	F1	0.87
	Accuracy	90.69% (+/- 2.54%)
D02 [un+adj-reg]	Precision	0.81
	Recall	0.79
	F1	0.80
	Accuracy	86.78% (+/- 1.99%)
D03 [adj+ly-reg]	Precision	0.98
	Recall	0.96
	F1	0.97
	Accuracy	97.80% (+/- 0.50%)
D04 [over+adj-reg]	Precision	0.94
	Recall	0.94
	F1	0.94
	Accuracy	96.17% (+/- 0.51%)
D05 [adj+ness-reg]	Precision	0.94
	Recall	0.96
	F1	0.95
	Accuracy	96.56% (+/- 1.29%)
D06 [re+verb-reg]	Precision	0.93
	Recall	0.89
	F1	0.91
	Accuracy	94.01% (+/- 0.64%)
D07 [verb+able-reg]	Precision	0.92
	Recall	0.97
	F1	0.94
	Accuracy	96.21% (+/- 1.19%)
D08 [verb+er-irreg]	Precision	0.91
	Recall	0.98
	F1	0.94
	Accuracy	95.93% (+/- 1.21%)
D09 [verb+tion-irreg]	Precision	0.97
	Recall	0.99
	F1	0.98
	Accuracy	98.60% (+/- 0.45%)
D10 [verb+ment-irreg]	Precision	0.96
	Recall	0.99
	F1	0.97
	Accuracy	98.10% (+/- 0.73%)

Table B.13: Classification results for CNN BATS Inflectional Morphology and Derivational Morphology - GloVe dimension: 50.

Class name	Metrics	Values
E01 [country - capital]	Precision	1.00
	Recall	1.00
	F1	1.00
	Accuracy	99.83% (+/- 0.19%)
E02 [country - language]	Precision	0.93
	Recall	0.98
	F1	0.96
	Accuracy	97.08% (+/- 0.38%)
E03 [UK-city - county]	Precision	0.84
	Recall	0.75
	F1	0.78
	Accuracy	86.63% (+/- 3.48%)
E04 [name - nationality]	Precision	0.84
	Recall	0.97
	F1	0.90
	Accuracy	92.79% (+/- 1.58%)
E05 [name - occupation]	Precision	0.93
	Recall	0.92
	F1	0.93
	Accuracy	95.06% (+/- 0.75%)
E06 [animal - young]	Precision	0.82
	Recall	0.80
	F1	0.80
	Accuracy	87.13% (+/- 1.42%)
E07 [animal - sound]	Precision	0.88
	Recall	0.84
	F1	0.86
	Accuracy	90.91% (+/- 1.28%)
E08 [animal - shelter]	Precision	0.80
	Recall	0.73
	F1	0.76
	Accuracy	84.93% (+/- 0.62%)
E09 [things - color]	Precision	0.81
	Recall	0.79
	F1	0.79
	Accuracy	86.49% (+/- 0.56%)
E10 [male - female]	Precision	0.94
	Recall	0.92
	F1	0.93
	Accuracy	95.21% (+/- 1.03%)

Class name	Metrics	Values
L01	Precision	0.80
	Recall	0.81
	F1	0.80
	Accuracy	78.70% (+/- 0.32%)
L02 [hypernyms - misc]	Precision	0.78
	Recall	0.75
	F1	0.76
	Accuracy	84.31% (+/- 0.24%)
L03 [hyponyms - misc]	Precision	0.93
	Recall	0.92
	F1	0.92.5
	Accuracy	92.02% (+/- 0.55%)
L04 [meronyms - substance]	Precision	0.61
	Recall	0.68
	F1	0.72
	Accuracy	82.39% (+/- 1.04%)
L05 [meronyms - member]	Precision	0.76
	Recall	0.71
	F1	0.73
	Accuracy	82.61% (+/- 0.85%)
L06 [meronyms - part]	Precision	0.62
	Recall	0.50
	F1	0.53
	Accuracy	76.23% (+/- 3.14%)
L07	Precision	0.81
	Recall	0.77
	F1	0.79
	Accuracy	86.27% (+/- 0.58%)
L08	Precision	0.62
	Recall	0.56
	F1	0.57
	Accuracy	73.48% (+/- %)
L09 [antonyms - gradable]	Precision	0.82
	Recall	0.82
	F1	0.82
	Accuracy	88.20% (+/- 0.23%)
L10 [antonyms - binary]	Precision	0.56
	Recall	0.46
	F1	0.50
	Accuracy	69.81% (+/- 0.51%)

Table B.14: Classification results for CNN on BATS Encyclopedic Semantics and Lexicographic Semantics - GloVe dimension: 50.

Appendix C. Detailed results on DiffVec dataset for classification

We present here the results for SVM, Random Forest (RF), NN 30 epochs and CNN on DiffVec using GloVe dimension 100 as explained in Subsection 5.1. The results for NN 30 epochs are very close to 100 epochs, so we only present the results for 30 epochs. For all algorithms, we exclude class 13 and 36 as they are too big for ML-based algorithms. For SVM, we also exclude classes 14 and 16 as they are too big for SVM with polynomial kernel (3 degrees of freedom).

Class	Metrics	SVM	RF	NN	CNN
1	P	0.50	0.45	0.41	0.19
	R	1.00	0.12	0.60	0.17
	F1	0.67	0.19	0.48	0.17
	Acc	66.67%	67.92%	58.75%	53.33%
2	P	0.62	1.00	0.99	0.85
	R	1.00	1.00	1.00	0.75
	F1	0.76	1.00	0.99	0.79
	Acc	79.39%	99.91%	99.56%	87.12%
3	P	0.72	1.00	0.98	0.96
	R	1.00	1.00	0.99	0.85
	F1	0.84	1.00	0.98	0.90
	Acc	86.80%	100.00%	98.99%	93.90%
4	P	0.50	0.88	0.86	0.60
	R	1.00	0.87	0.87	0.42
	F1	0.67	0.87	0.87	0.48
	Acc	66.60%	91.53%	91.02%	71.34%
5	P	0.79	1.00	0.99	0.95
	R	1.00	1.00	1.00	0.93
	F1	0.88	1.00	0.99	0.94
	Acc	90.87%	99.90%	99.48%	95.94%
6	P	0.61	0.86	0.85	0.75
	R	0.97	0.85	0.86	0.84
	F1	0.75	0.86	0.85	0.79
	Acc	78.04%	90.62%	90.20%	85.00%
7	P	0.50	0.81	0.80	0.64
	R	0.99	0.79	0.80	0.51
	F1	0.67	0.80	0.80	0.55
	Acc	67.22%	86.80%	86.50%	73.89%
8	P	0.50	1.00	0.94	0.73
	R	1.00	0.99	0.91	0.53
	F1	0.67	1.00	0.93	0.60
	Acc	66.97%	99.69%	95.29%	77.24%
9	P	0.58	1.00	0.98	0.82
	R	1.00	1.00	1.00	0.75
	F1	0.73	1.00	0.99	0.76
	Acc	75.78%	99.92%	99.30%	85.78%
10	P	0.59	1.00	0.98	0.95
	R	1.00	0.99	1.00	0.90
	F1	0.74	0.99	0.99	0.92
	Acc	76.49%	99.63%	99.19%	94.87%
11	P	0.74	1.00	0.97	0.89
	R	1.00	0.98	0.99	0.93
	F1	0.85	0.99	0.98	0.91
	Acc	88.09%	99.34%	98.89%	93.80%
12	P	0.99	1.00	0.99	0.76
	R	1.00	0.99	1.00	0.88
	F1	0.99	0.99	0.99	0.81
	Acc	99.56%	99.65%	99.59%	86.49%
14	P	Excluded	0.98	0.98	0.74
	R		0.97	0.98	0.70
	F1		0.97	0.98	0.72
	Acc		98.19%	98.61%	81.92%
15	P	0.75	0.67	0.78	1.00
	R	1.00	0.20	0.96	0.82
	F1	0.83	0.29	0.85	0.86
	Acc	83.33%	72.92%	88.33%	94.17%
16	P	Excluded	0.99	0.99	0.60
	R		0.98	0.99	0.74
	F1		0.98	0.99	0.65
	Acc		98.99%	99.27%	74.25%
17	P	0.84	1.00	1.00	0.95
	R	1.00	1.00	1.00	0.89
	F1	0.91	1.00	1.00	0.92
	Acc	93.64%	100.00%	99.89%	94.74%
18	P	0.96	1.00	1.00	0.92
	R	1.00	1.00	1.00	0.89
	F1	0.98	1.00	1.00	0.91
	Acc	98.50%	100.00%	99.88%	93.78%
19	P	0.56	1.00	0.98	0.76
	R	1.00	1.00	0.99	0.67
	F1	0.71	1.00	0.99	0.71
	Acc	73.22%	99.90%	99.19%	81.47%
20	P	0.88	1.00	0.98	0.93
	R	1.00	0.97	0.98	0.96
	F1	0.93	0.98	0.98	0.95
	Acc	95.11%	98.89%	98.57%	96.40%
21	P	0.93	1.00	0.99	0.92
	R	1.00	0.99	1.00	1.00
	F1	0.96	1.00	0.99	0.95
	Acc	97.55%	99.71%	99.40%	96.75%
22	P	0.52	1.00	0.95	0.82
	R	1.00	1.00	0.93	0.71
	F1	0.69	1.00	0.94	0.75
	Acc	69.44%	99.88%	96.12%	84.76%
23	P	0.50	1.00	0.98	0.73
	R	0.99	0.99	0.99	0.44
	F1	0.67	1.00	0.98	0.53
	Acc	67.14%	99.74%	98.77%	75.37%
24	P	0.64	1.00	0.99	0.89
	R	1.00	1.00	1.00	0.80
	F1	0.78	1.00	0.99	0.84
	Acc	80.79%	99.87%	99.49%	89.66%
25	P	0.63	1.00	0.98	0.79
	R	0.99	0.99	1.00	0.76
	F1	0.77	1.00	0.99	0.78
	Acc	80.26%	99.71%	99.11%	85.28%

Table C.15: Classification results for ML-based algorithms on DiffVec (class 1 to 12, 14 to 25) - GloVe dimension: 100.

Class	Metrics	SVM	RF	NN	CNN
26	P	0.50	1.00	0.94	0.66
	R	1.00	1.00	0.88	0.63
	F1	0.67	1.00	0.91	0.63
	Acc	66.67%	99.94%	94.34%	76.44%
27	P	0.62	1.00	0.98	0.96
	R	1.00	1.00	1.00	0.83
	F1	0.77	1.00	0.99	0.88
	Acc	79.59%	99.87%	99.37%	92.74%
28	P	0.88	1.00	0.98	0.98
	R	1.00	0.99	1.00	0.99
	F1	0.94	1.00	0.99	0.99
	Acc	95.40%	99.78%	99.15%	99.16%
29	P	0.52	0.86	0.85	0.77
	R	1.00	0.85	0.82	0.63
	F1	0.68	0.85	0.83	0.68
	Acc	68.62%	90.15%	89.10%	80.99%
30	P	0.50	1.00	0.98	0.83
	R	1.00	0.99	0.99	0.73
	F1	0.67	1.00	0.99	0.77
	Acc	67.21%	99.74%	99.04%	86.04%
31	P	0.50	1.00	0.98	0.46
	R	1.00	0.99	0.99	0.41
	F1	0.66	1.00	0.98	0.42
	Acc	66.54%	99.72%	98.83%	64.90%
32	P	0.50	0.83	0.80	0.69
	R	0.99	0.81	0.82	0.64
	F1	0.66	0.82	0.81	0.66
	Acc	66.38%	88.24%	87.11%	78.24%
33	P	0.99	1.00	1.00	1.00
	R	1.00	1.00	1.00	1.00
	F1	0.99	1.00	1.00	1.00
	Acc	99.19%	100.00%	99.93%	99.79%
34	P	0.96	1.00	1.00	0.99
	R	1.00	1.00	1.00	0.99
	F1	0.98	1.00	1.00	0.99
	Acc	98.74%	100.00%	99.93%	99.33%
35	P	0.98	1.00	1.00	0.99
	R	1.00	1.00	1.00	0.99
	F1	0.99	1.00	1.00	0.99
	Acc	99.17%	100.00%	99.90%	99.58%

Table C.16: Classification results for ML-based algorithms on DiffVec (class 26 to 35) - GloVe dimension: 100.