An Analogical Learner for Morphological Analysis

Nicolas Stroppa & François Yvon
GET/ENST & LTCI, UMR 5141
46 rue Barrault, 75013 Paris, France
{stroppa,yvon}@enst.fr

Abstract

Analogical learning is based on a two-step inference process: (i) computation of a structural mapping between a new and a memorized situation; (ii) transfer of knowledge from the known to the unknown situation. This approach requires the ability to search for and exploit such mappings, hence the need to properly define analogical relationships, and to efficiently implement their computation.

In this paper, we propose a unified definition for the notion of (formal) analogical proportion, which applies to a wide range of algebraic structures. We show that this definition is suitable for learning in domains involving large databases of structured data, as is especially the case in Natural Language Processing (NLP). We then present experimental results obtained on two morphological analysis tasks which demonstrate the flexibility and accuracy of this approach.

1 Introduction

Analogical learning (Gentner et al., 2001) is based on a two-step inductive process. The first step consists in the construction of a *structural* mapping between a new instance of a problem and solved instances of the same problem. Once this mapping is established, solutions for the new instance can be induced, based on one or several analogs. The implementation of this kind of inference process requires techniques for searching for, and reasoning with, structural mappings, hence the need to properly define the notion of analogical relationships and to efficiently implement their computation.

In Natural Language Processing (NLP), the typical dimensionality of databases, which are made up of hundreds of thousands of instances, makes the search for complex structural mappings a very challenging task. It is however possible to take advantage of the specific nature of linguistic data to work around this problem. Formal (surface) analogical relationships between linguistic representations are often a good sign of deeper analogies: a surface similarity between the word strings *write* and *writer* denotes a deeper (semantic) similarity between the related concepts. Surface similarities can of course be misleading. In order to minimize such confusions, one can take advantage of other specificities of linguistic data: (i) their systemic organization in (pseudo)-paradigms, and (ii) their high level of redundancy. In a large lexicon, we can indeed expect to find many instances of pairs like *write*-*writer*: for instance *read*-*reader*, *review*-*reviewer*...

Complementing surface analogies with statistical information thus has the potential to make the search problem tractable, while still providing with many good analogs. Various attempts have been made to use surface analogies in various contexts: automatic word pronunciation (Yvon, 1999), morphological analysis (Lepage, 1999a; Pirrelli and Yvon, 1999) and syntactical analysis (Lepage, 1999b). These experiments have mainly focused on linear represen-

Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL), pages 120–127, Ann Arbor, June 2005.

tations of linguistic data, taking the form of finite sequences of symbols, using a restrictive and sometimes *ad-hoc* definition of the notion of an analogy.

The first contribution of this paper is to propose a general definition of formal analogical proportions for algebraic structures commonly used in NLP: attribute-value vectors, words on finite alphabets and labeled trees. The second contribution is to show how these formal definitions can be used within an instance-based learning framework to learn morphological regularities.

This paper is organized as follows. In Section 2, our interpretation of analogical learning is introduced and related to other models of analogical learning and reasoning. Section 3 presents a general algebraic framework for defining analogical proportions as well as its instantiation to the case of words and labeled trees. This section also discusses the algorithmic complexity of the inference procedure. Section 4 reports the results of experiments aimed at demonstrating the flexibility of this model and at assessing its generalization performance. We conclude by discussing current limitations of this model and by suggesting possible extensions.

2 Principles of analogical learning

2.1 Analogical reasoning

The ability to identify analogical relationships between what looks like unrelated situations, and to use these relationships to solve complex problems, lies at the core of human cognition (Gentner et al., 2001). A number of models of this ability have been proposed, based on symbolic (e.g. (Falkenheimer and Gentner, 1986; Thagard et al., 1990; Hofstadter and the Fluid Analogies Research group, 1995)) or subsymbolic (e.g. (Plate, 2000; Holyoak and Hummel, 2001)) approaches. The main focus of these models is the dynamic process of analogy making, which involves the identification of a structural mappings between a memorized and a new situation. Structural mapping relates situations which, while being apparently very different, share a set of common high-level relationships. The building of a structural mapping between two situations utilizes several subparts of their descriptions and the relationships between them.

Analogy-making seems to play a central role in our reasoning ability; it is also invoked to explain some human skills which do not involve any sort of conscious reasoning. This is the case for many tasks related to the perception and production of language: lexical access, morphological parsing, word pronunciation, etc. In this context, analogical models have been proposed as a viable alternative to rule-based models, and many implementation of these low-level analogical processes have been proposed such as decision trees, neural networks or instance-based learning methods (see e.g. (Skousen, 1989; Daelemans et al., 1999)). These models share an acceptation of analogy which mainly relies on surface *similarities* between instances.

Our learner tries to bridge the gap between these approaches and attempts to remain faithful to the idea of structural analogies, which prevails in the AI literature, while also exploiting the intuitions of large-scale, instance-based learning models.

2.2 Analogical learning

We consider the following supervised learning task: a learner is given a set $\mathcal{S}$ of training instances $\{X_1, \ldots, X_n\}$ independently drawn from some unknown distribution. Each instance X_i is a vector containing m features: $\langle X_{i1}, \ldots, X_{im} \rangle$. Given $\mathcal{S}$, the task is to predict the missing features of partially informed new instances. Put in more standard terms, the set of known (resp. unknown) features for a new value X forms the *input space* (resp. *output space*): the projections of X onto the input (resp. output) space will be denoted $I(X)$ (resp. $O(X)$). This setting is more general than the simpler classification task, in which only one feature (the class label) is unknown, and covers many other interesting tasks.

The inference procedure can be sketched as follows: training examples are simply stored for future use; no generalization (abstraction) of the data is performed, which is characteristic of *lazy learning* (Aha, 1997). Given a new instance X, we identify formal analogical proportions involving X in the input space; known objects involved in these proportions are then used to infer the missing features.

An analogical proportion is a relation involving four objects A, B, C and D, denoted by $A : B :: C : D$ and which reads A *is to* B *as* C *is to* D. The definition and computation of these proportions are studied in Section 3. For the moment, we contend that it is possible to construct analogical proportions between (possibly partially informed) objects in $\mathcal{S}$. Let $I(X)$ be a partially described object not seen during training. The analogical inference process is formalized as:

1. Construct the set $\mathcal{T}(X) \subset \mathcal{S}^3$ defined as:

$$\mathcal{T}(X) = \{(A, B, C) \in \mathcal{S}^3 \mid I(A) : I(B) :: I(C) : I(X)\}$$

2. For each $(A, B, C) \in \mathcal{T}(X)$, compute hypotheses $\widehat{O(X)}$ by solving the equation:

$$\widehat{O(X)} = O(A) : O(B) :: O(C) :?$$

This inference procedure shows lots of similarities with the k-nearest neighbors classifier (k-NN) which, given a new instance, (i) searches the training set for close neighbors, (ii) compute the unknown class label according to the neighbors' labels. Our model, however, does not use any metric between objects: we only rely on the definition of analogical proportions, which reveal systemic, rather than superficial, similarities. Moreover, inputs and outputs are regarded in a symmetrical way: outputs are not restricted to a set of labels, and can also be structured objects such as sequences. The implementation of the model still has to address two specific issues.

- When exploring $\mathcal{S}^3$, an exhaustive search evaluates $|\mathcal{S}|^3$ triples, which can prove to be intractable. Moreover, objects in $\mathcal{S}$ may be unequally relevant, and we might expect the search procedure to treat them accordingly.

- Whenever several competing hypotheses are proposed for $\widehat{O(X)}$, a ranking must be performed. In our current implementation, hypotheses are ranked based on frequency counts.

These issues are well-known problems for k-NN classifiers. The second one does not appear to be critical and is usually solved based on a majority rule. In contrast, a considerable amount of effort has been devoted to reduce and optimize the search process, via editing and condensing methods, as studied e.g. in (Dasarathy, 1990; Wilson and Martinez, 2000). Proposals for solving this problem are discussed in Section 3.4.

3 An algebraic framework for analogical proportions

Our inductive model requires the availability of a device for computing analogical proportions on feature vectors. We consider that an analogical proportion holds between four feature vectors when the proportion holds for all components. In this section, we propose a unified algebraic framework for defining analogical proportions between individual features. After giving the general definition, we present its instantiation for two types of features: words over a finite alphabet and sets of labelled trees.

3.1 Analogical proportions

Our starting point will be analogical proportions in a set U, which we define as follows: $\forall x, y, z, t \in U, x : y :: z : t$ if and only if either $x = y$ and $z = t$ or $x = z$ and $y = t$. In the sequel, we assume that U is additionally provided with an associative internal composition law $\oplus$, which makes $(U, \oplus)$ a semigroup. The generalization of proportions to semigroups involves two key ideas: the *decomposition* of objects into smaller parts, subject to *alternation constraints*. To formalize the idea of decomposition, we define the *factorization* of an element u in U as:

Definition 1 (Factorization)
A factorization *of* $u \in U$ *is a sequence* $u_1 \ldots u_n$, *with* $\forall i, u_i \in U$, *such that:* $u_1 \oplus \ldots \oplus u_n = u$. *Each term* u_i *is a* factor *of* u.

The alternation constraint expresses the fact that analogically related objects should be made of alternating factors: for $x : y :: z : t$ to hold, each factor in x should be found alternatively in y and in z. This yields a first definition of analogical proportions:

Definition 2 (Analogical proportion)
$(x, y, z, t) \in U$ *form an* analogical proportion, *denoted by* $x : y :: z : t$ *if and only if there exists some factorizations* $x_1 \oplus \ldots \oplus x_d = x$, $y_1 \oplus \ldots \oplus y_d = y$, $z_1 \oplus \ldots \oplus z_d = z$, $t_1 \oplus \ldots \oplus t_d = t$ *such that* $\forall i, (y_i, z_i) \in \{(x_i, t_i), (t_i, x_i)\}$. *The smallest* d *for which such factorizations exist is termed the* degree *of the analogical proportion.*

This definition is valid for any semigroup, and *a fortiori* for any richer algebraic structure. Thus, it readily applies to the case of groups, vector spaces, free monoids, sets and attribute-value structures.

3.2 Words over Finite Alphabets

3.2.1 Analogical Proportions between Words

Let Σ be a finite alphabet. $\Sigma^\star$ denotes the set of finite sequences of elements of Σ, called *words* over Σ. $\Sigma^\star$, provided with the concatenation operation . is a free monoid whose identity element is the empty word ε. For $w \in \Sigma^\star$, $w(i)$ denotes the i^{th} symbol in w. In this context, definition (2) can be re-stated as:

Definition 3 (Analogical proportion in ($\Sigma^\star$,.)) $(x, y, z, t) \in \Sigma^\star$ *form an analogical proportion, denoted by* $x : y :: z : t$ *if and only if there exists some integer* d *and some factorizations* $x_1 \dots x_d = x$, $y_1 \dots y_d = y$, $z_1 \dots z_d = z$, $t_1 \dots t_d = t$ *such that* $\forall i, (y_i, z_i) \in \{(x_i, t_i), (t_i, x_i)\}$.

An example of analogy between words is:

$$viewing : reviewer :: searching : researcher$$

with $x_1 = \epsilon$, $x_2 = view$, $x_3 = ing$ and $t_1 = re$, $t_2 = search$, $t_3 = er$. This definition generalizes the proposal of (Lepage, 1998). It does not ensure the existence of a solution to an analogical equation, nor its uniqueness when it exists. (Lepage, 1998) gives a set of necessary conditions for a solution to exist. These conditions also apply here. In particular, if t is a solution of $x : y :: z :?$, then t contains, in the same relative order, all the symbols in y and z that are not in x. As a consequence, all solutions of an equation have the same length.

3.2.2 A Finite-state Solver

Definition (3) yields an efficient procedure for solving analogical equations, based on finite-state transducers. The main steps of the procedure are sketched here. A full description can be found in (Yvon, 2003). To start with, let us introduce the notions of *complementary set* and *shuffle product*.

Complementary set If v is a subword of w, the *complementary set* of v with respect to w, denoted by $w\backslash v$ is the set of subwords of w obtained by removing from w, in a left-to-right fashion, the symbols in v. For example, *eea* is a complementary subword of *xmplr* with respect to *exemplar*. When v is not a subword of w, $w\backslash v$ is empty. This notion can be generalized to any regular language.

The complementary set of v with respect to w is a regular set: it is the output language of the finite-state transducer T_w (see Figure 1) for the input v.

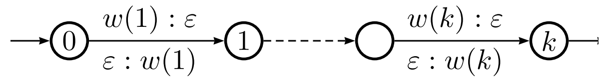

Figure 1: The transducer T_w computing complementary sets wrt w.

Shuffle The *shuffle* $u \bullet v$ of two words u and v is introduced e.g. in (Sakarovitch, 2003) as follows:

$$u \bullet v = \{u_1 v_1 u_2 v_2 \dots u_n v_n, \text{ st. } u_i, v_i \in \Sigma^\star, \\ u_1 \dots u_n = u, v_1 \dots v_n = v\}$$

The shuffle of two words u and v contains all the words w which can be composed using all the symbols in u and v, subject to the condition that if a precedes b in u (or in v), then it precedes b in w. Taking, for instance, $u = abc$ and $v = def$, the words $abcdef$, $abdefc$, $adbecf$ are in $u \bullet v$; this is not the case with $abefcd$. This operation generalizes straightforwardly to languages. The shuffle of two regular languages is regular (Sakarovitch, 2003); the automaton A, computing $K \bullet L$, is derived from the automata $A_K = (\Sigma, Q_K, q_K^0, F_K, \delta_K)$ and $A_L = (\Sigma, Q_L, q_L^0, F_L, \delta_L)$ recognizing respectively K and L as the product automata $A = (\Sigma, Q_K \times Q_L, (q_K^0, q_L^0), F_K \times F_L, \delta)$, where δ is defined as: $\delta((q_K, q_L), a) = (r_K, r_L)$ if and only if either $\delta_K(q_K, a) = r_K$ and $q_L = r_L$ or $\delta_L(q_L, a) = r_L$ and $q_K = r_K$.

The notions of complementary set and shuffle are related through the following property, which is a direct consequence of the definitions.

$$w \in u \bullet v \Leftrightarrow u \in w\backslash v$$

Solving analogical equations The notions of shuffle and complementary sets yield another characterization of analogical proportion between words, based on the following proposition:

Proposition 1.

$$\forall x, y, z, t \in \Sigma^\star, x : y :: z : t \Leftrightarrow x \bullet t \cap y \bullet z \neq \emptyset$$

An analogical proportion is thus established if the symbols in x and t are also found in y and z, and appear in the same relative order. A corollary follows:

Proposition 2.

$$t \text{ is a solution of } x : y :: z :? \Leftrightarrow t \in (y \bullet z)\backslash x$$

The set of solutions of an analogical equation $x : y :: z :?$ is a regular set, which can be computed with a finite-state transducer. It can also be shown that this analogical solver generalizes the approach based on edit distance proposed in (Lepage, 1998).

3.3 Trees

Labelled trees are very common structures in NLP tasks: they can represent syntactic structures, or terms in a logical representation of a sentence. To express the definition of analogical proportion between trees, we introduce the notion of substitution.

Definition 4 (Substitution)
A (single) substitution *is a pair* $(variable \leftarrow tree)$. *The application of the substitution* $(v \leftarrow t')$ *to a tree* t *consists in replacing each leaf of* t *labelled by* v *by the tree* t'. *The result of this operation is denoted:* $t(v \leftarrow t')$. *For each variable* v, *we define the binary operator* $\triangleleft_v$ *as* $t \triangleleft_v t' = t\,(v \leftarrow t')$.

Definition 2 can then be extended as:

Definition 5 (Analogical proportion (trees))
$(x, y, z, t) \in U$ *form an* analogical proportion, *denoted by* $x : y :: z : t$ *iff there exists some variables* $(v_1, \ldots, v_{n-1})$ *and some factorizations* $x_1 \triangleleft_{v_1} \ldots \triangleleft_{v_{n-1}} x_n = x$, $y_1 \triangleleft_{v_1} \ldots \triangleleft_{v_{n-1}} y_n = y$, $z_1 \triangleleft_{v_1} \ldots \triangleleft_{v_{n-1}} z_n = z$, $t_1 \triangleleft_{v_1} \ldots \triangleleft_{v_{n-1}} t_n = t$ *such that* $\forall i, (y_i, z_i) \in \{(x_i, t_i), (t_i, x_i)\}$.

An example of such a proportion is illustrated on Figure 2 with syntactic parse trees.

This definition yields an effective algorithm computing analogical proportions between trees (Stroppa and Yvon, 2005). We consider here a simpler heuristic approach, consisting in (i) linearizing labelled trees into parenthesized sequences of symbols and (ii) using the analogical solver for words introduced above. This approach yields a faster, albeit approximative algorithm, which makes analogical inference tractable even for large tree databases.

3.4 Algorithmic issues

We have seen how to compute analogical relationships for features whose values are words and trees.

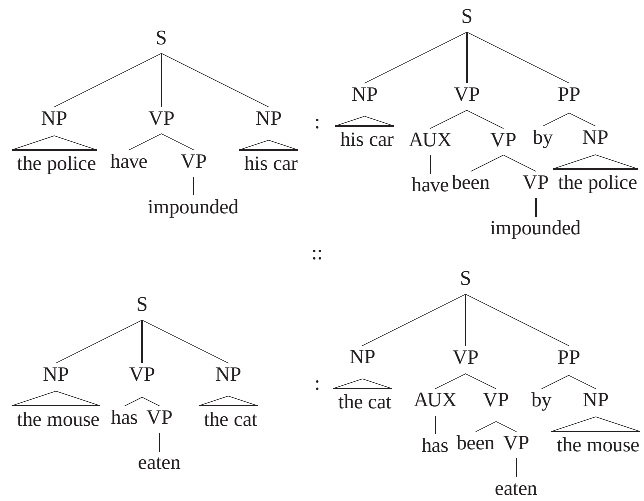

Figure 2: Analogical proportion between trees.

If we use, for trees, the solver based on tree linearizations, the resolution of an equation amounts, in both cases, to solving analogies on words.

The learning algorithm introduced in Section 2.2 is a two-step procedure: a search step and a transfer step. The latter step only involves the resolution of (a restricted number of) analogical equations. When x, y and z are known, solving $x : y :: z :?$ amounts to computing the output language of the transducer representing $(y \bullet z)\backslash x$: the automaton for this language has a number of states bounded by $|x| \times |y| \times |z|$. Given the typical length of words in our experiments, and given that the worst-case exponential bound for determinizing this automaton is hardly met, the solving procedure is quite efficient.

The problem faced during the search procedure is more challenging: given x, we need to retrieve all possible triples (y, z, t) in a finite set L such that $x : y :: z : t$. An exhaustive search requires the computation of the intersection of the finite-state automaton representing the output language of $(L \bullet L)\backslash x$ with the automaton for L. Given the size of L in our experiments (several hundreds of thousands of words), a complete search is intractable and we resort to the following heuristic approach.

L is first split into K bins $\{L_1, ..., L_K\}$, with $|L_i|$ small with respect to $|L|$. We then randomly select k bins and compute, for each bin L_i, the output language of $(L_i \bullet L_i)\backslash x$, which is then intersected with L: we thus only consider triples containing at least two words from the same bin. It has to be noted that the bins are not randomly constructed: training examples are grouped into inflectional or derivational families. To further speed up the search, we also impose an upper bound on the degree of proportions. All triples retrieved during these k partial searches are then merged and considered for the transfer step.

The computation of analogical relationships has been implemented in a generic analogical solver; this solver is based on Vaucanson, an automata manipulation library using high performance generic programming (Lombardy et al., 2003).

4 Experiments

4.1 Methodology

The main purpose of these experiments is to demonstrate the flexibility of the analogical learner. We considered two different supervised learning tasks, both aimed at performing the lexical analysis of isolated word forms. Each of these tasks represents a possible instantiation of the learning procedure introduced in Section 2.2.

The first experiment consists in computing one or several vector(s) of morphosyntactic features to be associated with a form. Each vector comprises the lemma, the part-of-speech, and, based on the part-of-speech, additional features such as number, gender, case, tense, mood, etc. An (English) input/output pair for this tasks thus looks like: input=*replying*; output={*reply*; `V-pp--`}, where the placeholder `'-'` denotes irrelevant features. Lexical analysis is useful for many applications: a POS tagger, for instance, needs to "guess" the possible part(s)-of-speech of unknown words (Mikheev, 1997). For this task, we use the definition of analogical proportions for "flat" feature vectors (see section 3.1) and for word strings (section 3.2). The training data is a list of fully informed lexical entries; the test data is a list of isolated word forms not represented in the lexicon. Bins are constructed based on inflectional families.

The second experiment consists in computing a morphological parse of unknown lemmas: for each input lemma, the output of the system is one or several parse trees representing a possible hierarchical decomposition of the input into (morphologically categorized) morphemes (see Figure 3). This kind of analysis makes it possible to reconstruct the series of morphological operations deriving a lemma, to compute its root, its part-of-speech, and to identify morpheme boundaries. This information is required, for instance, to compute the pronunciation of an unknown word; or to infer the compositional meaning of a complex (derived or compound) lemma. Bins gather entries sharing a common root.

input=*acrobatically*; output =

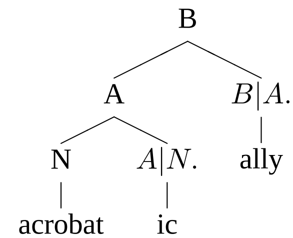

Figure 3: Input/output pair for task 2. Bound morphemes have a compositional type: $B|A.$ denotes a suffix that turns adjectives into adverbs.

These experiments use the English, German, and Dutch morphological tables of the CELEX database (Burnage, 1990). For task 1, these tables contain respectively 89 000, 342 000 and 324 000 different word forms, and the number of features to predict is respectively 6, 12, and 10. For task 2, which was only conducted with English lemma, the total number of different entries is 48 407.

For each experiment, we perform 10 runs, using 1 000 randomly selected entries for testing[1]. Generalization performance is measured as follows: the system's output is compared with the reference values (due to lexical ambiguity, a form may be associated in the database with several feature vectors or parse trees). Per instance *precision* is computed as the relative number of correct hypotheses, i.e. hypotheses which exactly match the reference: for task 1, all features have to be correct; for task 2, the parse tree has to be identical to the reference tree. Per instance *recall* is the relative number of reference values that were actually hypothesized. Precision and recall are averaged over the test set; numbers reported below are averaged over the 10 runs.

Various parameters affect the performance: k, the number of randomly selected bins considered during the search step (see Section 3.4) and d, the upper

[1] Due to lexical ambiguity, the number of tested instances is usually greater than 1 000.

bound of the degree of extracted proportions.

4.2 Experimental results

Experimental results for task 1 are given in Tables 1, 2 and 3. For each main category, two recall and precision scores are computed: one for the sole lemma and POS attributes (left column); and one for the lemma and all the morpho-syntactic features (on the right). In these experiments, parameters are set as follows: $k = 150$ and $d = 3$. As k grows, both recall and precision increase (up to a limit); $k = 150$ appears to be a reasonable trade-off between efficiency and accuracy. A further increase of d does not significantly improve accuracy: taking $d = 3$ or $d = 4$ yields very comparable results.

	Lemma + POS		Lemma + Features	
	Rec.	Prec.	Rec.	Prec.
Nouns	76.66	94.64	75.26	95.37
Verbs	94.83	97.14	94.79	97.37
Adjectives	26.68	72.24	27.89	87.67

Table 1: Results on task 1 for English

	Lemma + POS		Lemma + Features	
	Rec.	Prec.	Rec.	Prec.
Nouns	71.39	92.17	54.59	74.75
Verbs	96.75	97.85	93.26	94.36
Adjectives	91.59	96.09	90.02	95.33

Table 2: Results on task 1 for Dutch

	Lemma + POS		Lemma + Features	
	Rec.	Prec.	Rec.	Prec.
Nouns	93.51	98.28	77.32	81.70
Verbs	99.55	99.69	90.50	90.63
Adjectives	99.14	99.28	99.01	99.15

Table 3: Results on task 1 for German

As a general comment, one can note that high generalization performance is achieved for languages and categories involving rich inflectional paradigms: this is exemplified by the performance on all German categories. English adjectives, at the other end of this spectrum, are very difficult to analyze. A simple and effective workaround for this problem consists in increasing the size the sub-lexicons (L_i in Section 3.4) so as to incorporate in a given bin all the members of the same derivational (rather than inflectional) family. For Dutch, these results are comparable with the results reported in (van den Bosch and Daelemans, 1999), who report an accuracy of about 92% on the task of predicting the main syntactic category.

	Rec.	Prec.
Morphologically Complex	46.71	70.92
Others	17.00	46.86

Table 4: Results on task 2 for English

The second task is more challenging since the exact parse tree of a lemma must be computed. For morphologically complex lemmas (involving affixation or compounding), it is nevertheless possible to obtain acceptable results (see Table 4, showing that some derivational phenomena have been captured. Further analysis is required to assess more precisely the potential of this method.

From a theoretical perspective, it is important to realize that our model does not commit us to a morpheme-based approach of morphological processes. This is obvious in task 1; and even if task 2 aims at predicting a morphematic parse of input lemmas, this goal is achieved *without segmenting the input lemma into smaller units*. For instance, our learner parses the lemma *enigmatically* as: $[[[.N\, enigma][.A|N\, ical]]B|A.\, ly]$, that is without trying to decide to which morph the orthographic *t* should belong. In this model, input and output spaces are treated symmetrically and correspond to distinct levels of representation.

5 Discussion and future work

In this paper, we have presented a generic analogical inference procedure, which applies to a wide range of actual learning tasks, and we have detailed its instantiation for common feature types. Preliminary experiments have been conducted on two morphological analysis tasks and have shown promising generalization performance.

These results suggest that our main hypotheses are valid: (i) searching for triples is tractable even with databases containing several hundred of thousands instances; (ii) formal analogical proportions are a reliable sign of deeper analogies between linguistic entities; they can thus be used to devise flexible and effective learners for NLP tasks.

This work is currently being developed in various directions: first, we are gathering additional experimental results on several NLP tasks, to get a deeper understanding of the generalization capabilities of our analogical learner. One interesting issue, not addressed in this paper, is the integration of various forms of linguistic knowledge in the definition of analogical proportions, or in the specification of the search procedure. We are also considering alternative heuristic search procedures, which could improve or complement the approaches presented in this paper. A possible extension would be to define and take advantage of non-uniform distributions of training instances, which could be used both during the searching and ranking steps. We finally believe that this approach might also prove useful in other application domains involving structured data and are willing to experiment with other kinds of data.

References

David W. Aha. 1997. Editorial. *Artificial Intelligence Review*, 11(1-5):7–10. Special Issue on Lazy Learning.

Gavin Burnage. 1990. CELEX: a guide for users. Technical report, University of Nijmegen, Center for Lexical Information, Nijmegen.

Walter Daelemans, Antal Van Den Bosch, and Jakub Zavrel. 1999. Forgetting exceptions is harmful in language learning. *Machine Learning*, 34(1–3):11–41.

B.V. Dasarathy, editor. 1990. *Nearest neighbor (NN) Norms: NN Pattern Classification Techniques*. IEEE Computer Society Press, Los Alamitos, CA.

Brian Falkenheimer and Dedre Gentner. 1986. The structure-mapping engine. In *Proceedings of the meeting of the American Association for Artificial Intelligence (AAAI)*, pages 272–277.

Dedre Gentner, Keith J. Holyoak, and Boicho N. Konikov, editors. 2001. *The Analogical Mind*. The MIT Press, Cambridge, MA.

Douglas Hofstadter and the Fluid Analogies Research group, editors. 1995. *Fluid Concepts and Creative Analogies*. Basic Books.

Keith J. Holyoak and John E. Hummel. 2001. Understanding analogy within a biological symbol system. In Dedre Gentner, Keith J. Holyoak, and Boicho N. Konikov, editors, *The analogical mind*, pages 161–195. The MIT Press, Cambridge, MA.

Yves Lepage. 1998. Solving analogies on words: An algorithm. In *Proceedings of COLING-ACL '98*, volume 2, pages 728–735, Montréal, Canada.

Yves Lepage. 1999a. Analogy+tables=conjugation. In G. Friedl and H.G. Mayr, editors, *Proceedings of NLDB '99*, pages 197–201, Klagenfurt, Germany.

Yves Lepage. 1999b. Open set experiments with direct analysis by analogy. In *Proceedings of NLPRS '99*, volume 2, pages 363–368, Beijing, China.

Sylvain Lombardy, Raphaël Poss, Yann Régis-Gianas, and Jacques Sakarovitch. 2003. Introducing Vaucanson. In *Proceedings of CIAA 2003*, pages 96–107.

Andrei Mikheev. 1997. Automatic rule induction for unknown word guessing. *Computational Linguistics*, 23(3):405–423.

Vito Pirrelli and François Yvon. 1999. Analogy in the lexicon: a probe into analogy-based machine learning of language. In *Proceedings of the 6th International Symposium on Human Communication*, Santiago de Cuba, Cuba.

Tony A. Plate. 2000. Analogy retrieval and processing with distributed vector representations. *Expert systems*, 17(1):29–40.

Jacques Sakarovitch. 2003. *Eléments de théorie des automates*. Vuibert, Paris.

Royal Skousen. 1989. *Analogical Modeling of Language*. Kluwer, Dordrecht.

Nicolas Stroppa and François Yvon. 2005. Formal models of analogical relationships. Technical report, ENST, Paris, France.

Paul Thagard, Keith J. Holoyak, Greg Nelson, and David Gochfeld. 1990. Analog retrieval by constraint satisfaction. *Artificial Intelligence*, 46(3):259–310.

Antal van den Bosch and Walter Daelemans. 1999. Memory-based morphological processing. In *Proceedings of ACL*, pages 285–292, Maryland.

D. Randall Wilson and Tony R. Martinez. 2000. Reduction techniques for instance-based learning algorithms. *Machine Learning*, 38(3):257–286.

François Yvon. 1999. Pronouncing unknown words using multi-dimensional analogies. In *Proc. Eurospeech*, volume 1, pages 199–202, Budapest, Hungary.

François Yvon. 2003. Finite-state machines solving analogies on words. Technical report, ENST.