Conservative vs Set-driven Learning Functions for the Class k-valued

Christophe Costa Florêncio

UiL OTS, Utrecht University

Abstract

In (Buszkowski 1987, Buszkowski and Penn 1990) discovery procedures for CCGs were defined that accept a sequence of structures as input and yield a set of grammars.

In (Kanazawa 1998) it was shown that some of the classes based on these procedures are learnable from structures, and some functions were defined that learn these classes under certain restrictions on their behaviour. Two variants were defined, one that is conservative and one that is guaranteed to be set-driven. The question whether the latter is conservative was still left open, in this paper it is shown that it is not.

1 Introduction

In (Kanazawa 1998) the learnability (in the sense of identification in the limit, see (Gold 1967)) of certain classes of *classical categorial grammars* was studied. These classes were first proposed in (Buszkowski 1987, Buszkowski and Penn 1990) and are based on the unification of categorial types.

Among other things it was shown that some of these classes are learnable from structures (derivations), and are even learnable under quite severe restrictions on their behaviour. This paper solves an open question regarding the behaviour of certain proposed learning algorithms, in particular whether they are restricted to *conservative* and *set-driven* behaviour.

Unless stated otherwise, anything in this paper that is not familiar to the reader is defined in (Kanazawa 1998).

1.1 Set-drivenness, Conservativity and Other Constraints

A learning function φ is said to be set-driven if for every input sequence $\langle s_0, \dots s_n\rangle$, $\varphi(\langle s_0, \dots s_n\rangle) = \varphi(\{s_0, \dots s_n\})$. In other words, such a function is completely insensitive to repetition and order of presentation.

A learning function φ is said to be conservative if for every sentence $s_{n+1} \in \mathrm{L}\varphi(\langle s_0, \dots s_n\rangle)$, $\varphi(\langle s_0, \dots s_{n+1}\rangle) = \varphi(\langle s_0, \dots s_n\rangle)$. In other words, such a function only changes its hypothesis if data is encountered that was not predicted by its previous hypothesis.

Finally we mention:

- Responsive learning: the learning function must be defined for all sequences for languages for its class,
- Prudent learning: the learning function must not hypothesize grammars it is not prepared to learn,
- Consistent learning: the learning function must hypothesize grammars that generate all the data seen thus far.

Of all these constraints only responsiveness and prudence are not restrictive.

In (Kinber and Stephan 1995) it was shown that every class that is learnable by a set-driven function is learnable by a conservative function with linear memory. The inclusion is strict, i.e. there are conservatively learnable classes which cannot be learned by a set-driven function.

It was also shown that families learnable by a memory-limited function are exactly those learnable by a set-driven function.

2 Learning Functions Based on VG_k

Treating types as terms, the notions of unification and substitution apply naturally. They are defined for *grammars* as applying to the types occurring in these grammars. Let FL be the function that maps grammars to their structure languages.

The class $\mathcal{FL}_{k\text{-valued}}$ consists of all grammars that assign at most k types to any given word. The associated discovery procedure is VG_k, the learning function for this class is φ_{VG_k}.

Proposition 1 *If* $\sigma[G_1] \subseteq G_2$*, then* $\mathrm{FL}(G_1) \subseteq \mathrm{FL}(G_2)$.

Corollary 2 *If* $\sigma[G_1] \subseteq G_2$*, then* $\mathrm{L}(G_1) \subseteq \mathrm{L}(G_2)$.

Proposition 3 *If* $G \in \mathrm{VG}_k(D)$ *and* $T \in \mathrm{FL}(G)$*, then* $G \in \mathrm{VG}_k(D \cup \{T\})$.

We could call this ‘conservativity of grammar class’.

Definition 4 *Let* μ_{FL} *be a (computable) function that maps a non-empty finite set* $\mathcal{G}$ *of grammars to a grammar* $G \in \mathcal{G}$ *such that* $\mathrm{FL}(G)$ *is a minimal element of* $\{\mathrm{FL}(G) \mid G \in \mathcal{G}\}$.

Proposition 5 *For any finite set* $D \subset \Sigma^{\mathrm{F}}$*, if* $\mu_{\mathrm{FL}}(\mathrm{VG}_k(D))$ *is defined, then* $\mathrm{FL}(\mu_{\mathrm{FL}}(\mathrm{VG}_k(D)))$ *is a minimal element of* $\{L \in \mathcal{FL}_{k\text{-valued}} \mid D \subseteq L\}$.

Definition 6 *Let φ_{VG_k} be the learning function for $\langle \mathrm{CatG}, \Sigma^{\mathrm{F}}, \mathrm{FL} \rangle$ defined as follows:*

$$\begin{aligned}
&\varphi_{\mathrm{VG}_k}(\langle T_0 \rangle) = \mu_{\mathrm{FL}}(\mathrm{VG}_k(\{T_0\})),\\
&\varphi_{\mathrm{VG}_k}(\langle T_0, \ldots, T_{i+1} \rangle)\\
&\quad = \begin{cases} \varphi_{\mathrm{VG}_k}(\langle T_0, \ldots, T_i \rangle) & \text{if } T_{i+1} \in \mathrm{FL}(\varphi_{\mathrm{VG}_k}(\langle T_0, \ldots, T_i \rangle)),\\ \mu_{\mathrm{FL}}(\mathrm{VG}_k(\{T_0, \ldots, T_{i+1}\})) & \text{otherwise.} \end{cases}
\end{aligned}$$

This is a construction that is guaranteed to be conservative: it ignores input that fits into the current hypothesis.[1] Only if input is not compatible with the current hypothesis (i.e., is not in the structure language of the former output grammar), a new hypothesis is considered. Note that the learning function based on a μ-like function and some discovery procedure may not be *inherently* conservative.

Proposition 7

1. *φ_{VG_k} is responsive and consistent on $\mathcal{G}_{k\text{-valued}}$.*
2. *φ_{VG_k} is conservative.*
3. *φ_{VG_k} learns $\mathcal{G}_{k\text{-valued}}$ prudently.*

Theorem 8 *φ_{VG_k} learns $\mathcal{G}_{k\text{-valued}}$ from structures.*

The function φ_{VG_k} is not designed to be set-driven or even to learn order-independently. Kanazawa defines a set-driven learning function $\varphi^{\flat}_{\mathrm{VG}_k}$:

Definition 9

$$\varphi^{\flat}_{\mathrm{VG}_k}(\langle T_0, \ldots, T_i \rangle) = \mu_{\mathrm{FL}}(\mathrm{VG}_k(\{T_0, \ldots, T_i\})),$$

where μ_{FL} is defined as follows:

Definition 10 *Let μ_{FL} be a computable total function that maps a finite set $\mathcal{G}$ of grammars to the first element of*
$\{G \in \mathcal{G} \mid \mathrm{FL}(G)$ is a minimal element of $\mathrm{FL}(\mathcal{G})\}$ under the ordering $\prec$.

[1]Note that it is only ignored 'locally'. Once input does not fit, the input that was formerly ignored *is* taken into account when constructing a new hypothesis.

Here, $\prec$ is defined by[2]:

Definition 11 *Let $\prec$ be a computable well-order on CatG such that $G_1 \prec G_2$ whenever one of the following conditions holds:*

1. *size$(G_1) <$ size(G_2).*
2. *size$(G_1) =$ size(G_2) and $|\mathrm{Var}(G_1)| > |\mathrm{Var}(G_2)|$.*
3. *size$(G_1) =$ size(G_2) and $|\mathrm{Var}(G_1)| = |\mathrm{Var}(G_2)|$, then $G_1 \prec G_2$ by some arbitrary lexicographic ordering of grammars.*

The size of a grammar is defined as:

Definition 12 *For any grammar G, define the* size *of G, $\mathrm{size}(G)$, as follows:*

$$\mathrm{size}(G) = \sum_{c \in \Sigma} \sum_{G:\, c \mapsto A} |A|$$

where for each type A, $|A|$ is the number of symbol occurrences in A.

Lemma 13 *If $G_1 \sqsubseteq G_2$, then $\mathrm{size}(G_1) \leq \mathrm{size}(G_2)$.*

Lemma 14 *$G_1 \sqsubset G_2$ implies $G_1 \prec G_2$.*

Proposition 15 *Let $\langle T_i \rangle_{i \in \mathbb{N}}$ be an infinite sequence enumerating some $L \in \mathcal{FL}_{k\text{-valued}}$. Then $\varphi^{\flat}_{\mathrm{VG}_k}$ converges on $\langle T_i \rangle_{i \in \mathbb{N}}$ to the first element of*

$$\mathcal{G}_L = \{G \in \mathcal{G}_{k\text{-valued}} \mid \mathrm{FL}(G) = L\}$$

under the ordering $\prec$.

While $\varphi^{\flat}_{\mathrm{VG}_k}$ is set-driven, Kanazawa left it an open question whether it is conservative.

Our proof of non-conservativity of $\varphi^{\flat}_{\mathrm{VG}_k}$ was inspired by a footnote on page 102 of (Kanazawa 1998):

[2]Even though it is not clear why Kanazawa chose this particular ordering, this definition suggests the adapted version of μ_{FL} is intended to pick the 'simplest' (in an informal sense) grammar from a set.
There exists a learning strategy called *simplicity*. (See (Osherson, Stob and Weinstein 1986).) This strategies constrains learning functions not to conjecture grammars that are arbitrarily more complex than simpler alternatives for the same language. For any size measure and any bound on the size difference between conjecture and simpler alternative, this strategy severely restricts the class of learnable languages in the recursive case.

Although I have not had time to prove that $\varphi^{\flat}_{\mathrm{VG}_k}$ is indeed nonconservative, it is conceivable that the following sort of situation can obtain: $G_0 = \mu_{\mathrm{FL}}(\mathrm{VG}_k(D))$. $G_1 \prec G_0$, $G_1 \in \mathrm{VG}_k(D)$, there is no $G'_1 \in \mathrm{VG}_k(D)$ such that $G'_1 \sqsubset G_1$, but $\mathrm{FL}(G_1)$ is not minimal in $\mathrm{FL}(\mathrm{VG}_k(D))$. $G_0, G_1 \in \mathrm{VG}_k(D \cup \{T\})$, and $\mathrm{FL}(G_1)$ is minimal in $\mathrm{FL}(\mathrm{VG}_k(D \cup \{T\}))$.

To summarize, under the following conditions $\varphi^{\flat}_{\mathrm{VG}_k}$ is non-conservative:

1. $G_0, G_1 \in \mathrm{VG}_k(D)$, $\mathrm{FL}(G_0)$ is minimal, $G_1 \prec G_0$, $\varphi^{\flat}_{\mathrm{VG}_k}(D) = G_0$.

2. $G_0, G_1 \in \mathrm{VG}_k(D \cup \{T\})$, $\mathrm{FL}(G_0)$ is not minimal, $\varphi^{\flat}_{\mathrm{VG}_k}(D \cup \{T\}) = G_1$.

It turns out that such a situation can occur, and implies the following:

There is a finite (possibly empty) set of grammars $FG \subset \mathrm{VG}_k(D)$ such that all $G \in FG$, $\mathrm{FL}(G) \nsubseteq \mathrm{FL}(G_0)$ and $\mathrm{FL}(G_0) \nsubseteq \mathrm{FL}(G)$, and $G_0 \prec G$. For all $G \in \mathrm{VG}_k(D) - FG$, $\mathrm{FL}(G_0) \subseteq \mathrm{FL}(G)$.

Since $G_1 \prec G_0$ and $\mu_{\mathrm{FL}}(\mathrm{VG}_k(D)) = G_0$,$\neg(\mathrm{FL}(G_0) \subset \mathrm{FL}(G_1))$. Then, G_1 is not minimal in $\mathrm{VG}_k(D)$, so there is a grammar $G_2 \in \mathrm{VG}_k(D)$ such that $\mathrm{FL}(G_2) \subset \mathrm{FL}(G_1)$ and $G_0 \prec G_2$.

Moreover, G_2 cannot be in $\mathrm{VG}_k(D \cup \{T\})$. If it would, μ_{FL} would choose G_2 over G_1, so condition 2 could not be satisfied. So we have $\mathrm{FL}(G_2) \not\subset \mathrm{FL}(G_0)$, and $G_0 \prec G_2$, so $G_1 \prec G_0 \prec G_2$.

By Proposition 3, since $G_2 \in \mathrm{VG}_k(D)$ and $G_2 \notin \mathrm{VG}_k(D \cup \{T\})$, $T \notin \mathrm{FL}(G_2)$.

Since for any $G \in \mathrm{VG}_k(D)$, $D \subseteq \mathrm{FL}(G)$, $\{T\} \subset \mathrm{FL}(G_0)$, $\{T\} \subseteq \mathrm{FL}(G_1)$.

We have designed a set of grammars that fulfills all of the above conditions. Before we give the proof, the following definition will be convenient:

Definition 16 *Let* $\mathrm{UNI}(G)$ *denote the function that generates a finite set of grammars* $\mathcal{G}$ *given the grammar* G*, such that every member* G_i *of* $\mathcal{G}$ *is* $\sigma_i[G]$*, where for every* i *and* j*,* $i \neq j, \sigma_i \neq \sigma_j$*, and every* σ_i *is a result of unifying two types assigned to the same word in* G*.*

In other words, UNI *nondeterministically unifies all pairs of types in G that are under obligation to unify, and associates a new grammar with each of these pairs.*

Note that no $G_i \in \mathrm{UNI}(G)$ can be an alphabetic variant of G. The function $\mathrm{UNI}(G)$ is at the heart of the algorithms of Buszkowski and Penn.

Proposition 17 *For any* $G' \in \mathrm{UNI}(G), \mathrm{FL}(G) \subseteq \mathrm{FL}(G')$.

`Proof:` $G' \in \mathrm{UNI}(G)$ implies that there exists a substitution σ such that $\sigma[G](= G') \subseteq G$. By Proposition 1, $\mathrm{FL}(G) \subseteq \mathrm{FL}(G')$ follows. $\square$

Proposition 18 *The set-driven learning function* $\varphi^{\flat}_{\mathrm{VG}_k}$ *is non-conservative.*

`Proof:` By example. Let D be the following sample:

```
ba(fa(a,fa(b,fa(x,x))),g)                g
ba(fa(fa(y,y),fa(fa(y,y),fa(x,x))),g)    ba(a,e)
ba(b,e)                                  e
ba(ba(fa(z,z),a),j)                      j
ba(ba(fa(z,z),fa(w,w)),j)                ba(fa(y,y),c)
ba(fa(w,w),c)                            ba(d,e)
ba(ba(fa(z,z),d),j)                      ba(fa(a,fa(f,fa(x,x))),g)
ba(f,e)
```

It's useful to note that with a 2-valued grammar, given a structure of the form `fa(x,x)` (or `ba(x,x)`), `x` must be assigned the same types whenever it occurs. The discovery procedure $\mathrm{VG}_{k\text{-valued}}$ outputs the following three grammars.[3] Note that they differ only in the types assigned to `a`, `b`, `d`, `e`, and `f`.

G_1 :

a	$\mapsto$	$B/B, D\backslash E$
b	$\mapsto$	B/B
c	$\mapsto$	$(B/B)\backslash t, (D\backslash E)\backslash t$
d	$\mapsto$	$B/B, D\backslash E$
e	$\mapsto$	$(B/B)\backslash t, t$
f	$\mapsto$	B/B
g	$\mapsto$	$B\backslash t, t$
j	$\mapsto$	$E\backslash t, t$
w	$\mapsto$	$(D\backslash E)/W, W$
x	$\mapsto$	$B/X, X$
y	$\mapsto$	$(B/B)/Y, Y$
z	$\mapsto$	$D/Z, Z$

[3]This has been verified using the Prolog implementation from (Kanazawa 1998), more specifically with the predicate `valued_grammar/3`.

$$G_2 : \quad \begin{array}{lll}
\texttt{a} & \mapsto & B/C, D\backslash E \\
\texttt{b} & \mapsto & B/C, C/B \\
\texttt{c} & \mapsto & (B/B)\backslash t, (D\backslash E)\backslash t \\
\texttt{d} & \mapsto & B/C, D\backslash E \\
\texttt{e} & \mapsto & (B/C)\backslash t, t \\
\texttt{f} & \mapsto & B/C, C/B \\
\texttt{g} & \mapsto & B\backslash t, t \\
\texttt{j} & \mapsto & E\backslash t, t \\
\texttt{w} & \mapsto & (D\backslash E)/W, W \\
\texttt{x} & \mapsto & B/X, X \\
\texttt{y} & \mapsto & (B/B)/Y, Y \\
\texttt{z} & \mapsto & D/Z, Z
\end{array}$$

$$G_3 : \quad \begin{array}{lll}
\texttt{a} & \mapsto & B/C, D\backslash E \\
\texttt{b} & \mapsto & C/B, D\backslash E \\
\texttt{c} & \mapsto & (B/B)\backslash t, (D\backslash E)\backslash t \\
\texttt{d} & \mapsto & D\backslash E \\
\texttt{e} & \mapsto & (D\backslash E)\backslash t, t \\
\texttt{f} & \mapsto & C/B, D\backslash E \\
\texttt{g} & \mapsto & B\backslash t, t \\
\texttt{j} & \mapsto & E\backslash t, t \\
\texttt{w} & \mapsto & (D\backslash E)/W, W \\
\texttt{x} & \mapsto & B/X, X \\
\texttt{y} & \mapsto & (B/B)/Y, Y \\
\texttt{z} & \mapsto & D/Z, Z
\end{array}$$

Note that G_1 is the result of unifying C with B in G_2, and $\mathrm{FL}(G_1)$ properly includes $\mathrm{FL}(G_2)$. $\mathrm{FL}(G_2)$ and $\mathrm{FL}(G_3)$ are incomparable and $\mathrm{size}(G_2) > \mathrm{size}(G_3)$, so the set-driven learning function based on $\prec$ picks G_3 for this sample.

Now consider adding

```
ba(b,c)
```

to the above sample and see what grammars are output:

$$G'_1 : \quad \begin{array}{lcl}
\mathtt{a} & \mapsto & B/B, D\backslash E \\
\mathtt{b} & \mapsto & B/B \\
\mathtt{c} & \mapsto & (B/B)\backslash t, (D\backslash E)\backslash t \\
\mathtt{d} & \mapsto & B/B, D\backslash E \\
\mathtt{e} & \mapsto & (B/B)\backslash t, t \\
\mathtt{f} & \mapsto & B/B \\
\mathtt{g} & \mapsto & B\backslash t, t \\
\mathtt{j} & \mapsto & E\backslash t, t \\
\mathtt{w} & \mapsto & (D\backslash E)/W, W \\
\mathtt{x} & \mapsto & B/X, X \\
\mathtt{y} & \mapsto & (B/B)/Y, Y \\
\mathtt{z} & \mapsto & D/Z, Z
\end{array}$$

$$G'_2 : \quad \begin{array}{lcl}
\mathtt{a} & \mapsto & B/B, D\backslash E \\
\mathtt{b} & \mapsto & B/B, D\backslash E \\
\mathtt{c} & \mapsto & (B/B)\backslash t, (D\backslash E)\backslash t \\
\mathtt{d} & \mapsto & D\backslash E \\
\mathtt{e} & \mapsto & (D\backslash E)\backslash t, t \\
\mathtt{f} & \mapsto & B/B, D\backslash E \\
\mathtt{g} & \mapsto & B\backslash t, t \\
\mathtt{j} & \mapsto & E\backslash t, t \\
\mathtt{w} & \mapsto & (D\backslash E)/W, W \\
\mathtt{x} & \mapsto & B/X, X \\
\mathtt{y} & \mapsto & (B/B)/Y, Y \\
\mathtt{z} & \mapsto & D/Z, Z
\end{array}$$

$$G'_3 : \quad \begin{array}{lcl}
\mathtt{a} & \mapsto & B/B, D\backslash E \\
\mathtt{b} & \mapsto & B/B, D\backslash E \\
\mathtt{c} & \mapsto & (B/B)\backslash t, (D\backslash E)\backslash t \\
\mathtt{d} & \mapsto & B/B, D\backslash E \\
\mathtt{e} & \mapsto & (B/B)\backslash t, t \\
\mathtt{f} & \mapsto & B/B \\
\mathtt{g} & \mapsto & B\backslash t, t \\
\mathtt{j} & \mapsto & E\backslash t, t \\
\mathtt{w} & \mapsto & (D\backslash E)/W, W \\
\mathtt{x} & \mapsto & B/X, X \\
\mathtt{y} & \mapsto & (B/B)/Y, Y \\
\mathtt{z} & \mapsto & D/Z, Z
\end{array}$$

$$G'_4 : \begin{array}{lcl} \texttt{a} & \mapsto & B/C, D\backslash E \\ \texttt{b} & \mapsto & C/B, D\backslash E \\ \texttt{c} & \mapsto & (B/B)\backslash t, (D\backslash E)\backslash t \\ \texttt{d} & \mapsto & D\backslash E \\ \texttt{e} & \mapsto & (D\backslash E)\backslash t, t \\ \texttt{f} & \mapsto & C/B, D\backslash E \\ \texttt{g} & \mapsto & B\backslash t, t \\ \texttt{j} & \mapsto & E\backslash t, t \\ \texttt{w} & \mapsto & (D\backslash E)/W, W \\ \texttt{x} & \mapsto & B/X, X \\ \texttt{y} & \mapsto & (B/B)/Y, Y \\ \texttt{z} & \mapsto & D/Z, Z \end{array}$$

Note that G'_1 is the same as G_1, and G'_4 is the same as G_3. G'_2 is the result of unifying C with B in G'_4, and G'_3 is G'_1 plus one additional type assignment: $\texttt{b} \mapsto D\backslash E$. So $\mathrm{FL}(G'_2)$ properly includes $\mathrm{FL}(G'_4)$ and $\mathrm{FL}(G'_3)$ properly includes $\mathrm{FL}(G'_1)$. $\mathrm{FL}(G'_1)$ and $\mathrm{FL}(G'_4)$ are incomparable. But $\mathrm{size}(G'_1) < \mathrm{size}(G'_4)$, so G'_1, not G'_4 (= G_3), is the grammar picked by the set-driven learning function for this expanded sample. $\square$

3 Conclusions

It has been demonstrated that a learning function proposed in (Kanazawa 1998) is set-driven but not conservative. The proof is by example, and we feel that the techniques used for constructing a sample that forces the algorithm to display (un)wanted behaviour are sufficiently general to be applicable to a wide range of language classes.

4 Acknowledgments

I am indebted to Makoto Kanazawa for his constructive criticism.

References

Buszkowski, W.(1987), Discovery procedures for categorial grammars, *in* E. Klein and J. van Benthem (eds), *Categories, Polymorphism and Unification*, University of Amsterdam.

Buszkowski, W. and Penn, G.(1990), Categorial grammars determined from linguistic data by unification, *Studia Logica* **49**, 431–454.

Gold, E. M.(1967), Language identification in the limit, *Information and Control* **10**, 447–474.

Kanazawa, M.(1998), *Learnable Classes of Categorial Grammars*, CSLI Publications, Stanford University.

Kinber, E. and Stephan, F.(1995), Language learning from texts: Mind-changes, limited memory, and monotonicity, *Information and Computation* **123**(2), 224–241.

Osherson, D. N., Stob, M. and Weinstein, S.(1986), *Systems that Learn: An Introduction to Learning Theory for Cognitive and Computer Scientists*, MIT Press, Cambridge, MA.