0.1 Strictly k-Local Stringsets

Here we present an algorithm and prove that it identifies the Strictly k-Local (SL_k) stringsets in the limit from positive data. The first proof of this result was presented by Garcia *et al.* (1990), though the Markovian principles underlying this result were understood in a statistical context much earlier. The learning scheme discussed there exemplifies more general ideas (Heinz, 2010; Heinz *et al.*, 2012).

The notion of *substring* is integral to SL stringsets. Formally, a string u is substring of string v ($u \trianglelefteq v$) provided there are strings $x, y \in \Sigma^*$ and $v = xuy$. Another term for substring is *factor*. So we also say that u is a factor of v. If u is of length k then we say u is a k-factor of v.

A stringset S is Strictly k-Local if and only if there is a number k such that for all strings $u_1, v_1, u_2, v_2, x \in \Sigma^*$ such that if $|x| = k$ and $u_1xv_1, u_2xv_2 \in S$ then $u_1xv_2 \in S$. We say S is closed under suffix substitution (Rogers and Pullum, 2011).

A theorem shows that every SL_k stringset S has a basis in a finite set of strings (Rogers and Pullum, 2011). These strings can be understood as *forbidden* substrings. Informally, this means any string s containing any one of the forbidden substrings is not in S. Conversely, any string s which does not contain any forbidden substring belongs to S.

The same theorem shows that a SL stringset S can be defined in terms of a finite set of *permissible* substrings. In this case, s belongs to S if and only if every one of its k-factors is permissible.

We formalize the above notions by first defining a function the $\texttt{factor}_k$, which extracts the substrings of length k present in a string, or those present in a set of strings. If a string s is of length less than k then $\texttt{factor}_k$ just returns s.

Formally, let $\texttt{factor}_k(s)$ equal $\{u \mid u \trianglelefteq s, |u| = k\}$ whenever $k \leq |s|$ and let $\texttt{factor}_k(s) = \{s\}$ whenever $|s| < k$. We expand the domain of this function to include sets of strings as follows: $\texttt{factor}_k(S) = \bigcup_{s \in S} \texttt{factor}_k(s)$.

To formally define SL_k grammars, we introduce the symbols $\rtimes$ and $\ltimes$, which denote left and right word boundaries, respectively. These symbols are introduced because we also want to be able to forbid specific strings at the beginning and ends of words, and traditionally strictly local stringsets were defined to make such distinctions (McNaughton and Papert, 1971). Then let a grammar G be a finite subset of $\texttt{factor}_k(\{\rtimes\}\Sigma^*\{\ltimes\})$.

The "language of the grammar" $L(G)$ is defined as the stringset $\{s \mid \texttt{factor}_k(\rtimes s \ltimes) \subseteq G\}$. We are going to be interested in the collection of stringsets SL_k, defined as those stringsets generated from grammars G with a longest string k. Formally,

$$\mathrm{SL}_k \stackrel{\text{def}}{=} \{S \mid G \subseteq \texttt{factor}_k(\{\rtimes\}\Sigma^*\{\ltimes\}), L(G) = S\} \ .$$

This is the collection C of learning targets.

For all $S \in \mathrm{SL}_k$, all presentations φ of S, and all time points $t \in \mathbb{N}$ define A as follows:

$$A\big(\varphi\langle t\rangle\big) = \begin{cases} \varnothing & \text{if } t = 0 \\ A(\varphi\langle t-1\rangle) \cup \texttt{factor}_k(\rtimes\varphi(t)\ltimes) & \text{otherwise} \end{cases}$$

One can prove that algorithm A identifies in the limit from positive data the collection of stringsets SL_k.

Exercise 1. Prove algorithm A identifies in the limit from positive data the collection of stringsets SL_k.

References

Garcia, Pedro, Enrique Vidal, and José Oncina. 1990. Learning locally testable languages in the strict sense. In *Proceedings of the Workshop on Algorithmic Learning Theory*, 325–338.

Heinz, Jeffrey. 2010. String extension learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, 897–906. Uppsala, Sweden: Association for Computational Linguistics.

Heinz, Jeffrey, Anna Kasprzik, and Timo Kötzing. 2012. Learning with lattice-structured hypothesis spaces. *Theoretical Computer Science* 457:111–127.

McNaughton, Robert, and Seymour Papert. 1971. *Counter-Free Automata*. MIT Press.

Rogers, James, and Geoffrey Pullum. 2011. Aural pattern recognition experiments and the subregular hierarchy. *Journal of Logic, Language and Information* 20:329–342.