A Universal Kernel for Learning Regular Languages

Leonid (Aryeh) Kontorovich
Department of Mathematics
Weizmann Institute of Science
Rehovot, Israel 76100

Abstract

We give a universal kernel that renders all the regular languages linearly separable. We are not able to compute this kernel efficiently and conjecture that it is intractable, but we do have an efficient ϵ-approximation.

1 Background

Since the advent of Support Vector Machines (SVMs), kernel methods have flourished in machine learning theory [7]. Formally, a *kernel* is a positive definite function from $\mathcal{X} \times \mathcal{X}$ to $\mathbb{R}$, which, via Mercer's theorem, endows an abstract set with the structure of a Hilbert space. Kernels provide both computational and theoretical power. The so-called *kernel trick*, when available, allows us to bypass computing the explicit embedding $\phi : \mathcal{X} \to \mathbb{R}^{\mathcal{F}}$ in feature space via the identity $K(x, y) = \langle \phi(x), \phi(y) \rangle$; this can lead to a considerable gain in efficiency. On a more conceptual level, imposing an inner product space structure on an abstract set allows us to harness the theoretical and computational utility of linear algebra and convex optimization.

A concrete example where kernel methods provide a palpable advantage over more direct approaches is that of learning finite automata from labeled strings. Indeed, the most obvious way to infer a DFA from such a sample is to build the smallest automaton that accepts all the positive strings and none of the negative ones. A straightforward "Occam's Razor" argument [4, Theorem 2.1] shows that with this strategy, a polynomial (in $1/\epsilon, 1/\delta$ and target automaton size) number of samples is sufficient to ensure a generalization error of no more than ϵ with confidence at least $1 - \delta$. Of course, there has to be a catch – finding the smallest automaton consistent with a set of accepted and rejected strings was shown to be NP-complete by Angluin [1] and Gold [3]; this was further strengthened in the hardness of approximation result of Pitt and Warmuth [6].

In [5], Kontorovich, Cortes and Mohri proposed an alternate framework for learning regular languages. Strings are embedded in a high-dimensional space and language induction is achieved by constructing a maximum-margin hyperplane. This hinges on every language in a family of interest being *linearly separable* under the embedding, and on the efficient computability of the kernel.

This line of research is continued in [2], where linear separability properties of rational kernels are investigated.

In this paper, we give a universal kernel that renders all the regular languages linearly separable. Any linearly separable language necessarily has a positive margin, and standard generalization guarantees apply; see [5] for details. We are not able to compute this kernel efficiently and conjecture that it is intractable, but we do have an efficient ϵ-approximation. Even with these limitations, it appears that the technique we propose is the first tool to tackle unsupervised learning of unrestricted regular languages.

2 Linearly separable concept classes

Let $\mathcal{C}$ be a countable concept class defined over a countable set $\mathcal{X}$. We will say that a concept $c \in \mathcal{C}$ is *finitely linearly separable* if there exists a mapping $\phi : \mathcal{X} \to \{0,1\}^{\mathbb{N}}$ and a weight vector $w \in \mathbb{R}^{\mathbb{N}}$, both with *finite support*, i.e., $\|w\|_0 < \infty$ and $\|\phi(x)\|_0 < \infty$ for all $x \in \mathcal{X}$, such that

$$c = \{x \in \mathcal{X} : \langle w, \phi(x)\rangle > 0\}.$$

The concept class $\mathcal{C}$ is said to be *finitely linearly separable* if all $c \in \mathcal{C}$ are finitely linearly separable under the same mapping ϕ.

Note that the condition $\|\phi(\cdot)\|_0 < \infty$ is important; otherwise, we could define the *embedding by concept*[1] $\phi : \mathcal{X} \to \{0,1\}^{\mathcal{C}}$

$$[\phi(x)]_c \quad = \quad \mathbb{1}_{\{x \in c\}}, \qquad c \in \mathcal{C}$$

and for any target $\hat{c} \in \mathcal{C}$,

$$w_c \quad = \quad \mathbb{1}_{\{c = \hat{c}\}}.$$

This construction trivially ensures that

$$\langle w, \phi(x)\rangle \quad = \quad \mathbb{1}_{\{x \in \hat{c}\}}, \qquad x \in \mathcal{X}$$

(another reason to require $\|\phi(\cdot)\|_0 < \infty$ is that it automatically makes the kernel $K(x,y) = \langle \phi(x), \phi(y)\rangle$ well-defined for all $x, y \in \mathcal{X}$).

Similarly, we disallow $\|w\|_0 = \infty$ due to the algorithmic impossibility of storing infinitely many numbers and also because it leads to the trivial construction, via *embedding by instance*:

$$[\phi(x)]_u \quad = \quad \mathbb{1}_{\{x = u\}}, \qquad u \in \mathcal{X},$$

and for any target $\hat{c} \in \mathcal{C}$,

$$w_u \quad = \quad \mathbb{1}_{\{u \in \hat{c}\}}.$$

This again ensures $\langle w, \phi(x)\rangle = \mathbb{1}_{\{x \in \hat{c}\}}$ without doing anything interesting or useful.

[1] Throughout this paper, we index vectors by integers or members of other countable sets, as dictated by convenience.

In light of the examples above, from now on when we speak of linear separability of a concept class, we shall always assume that $\mathcal{X}$ and $\mathcal{C}$ are countable and that w and $\phi(\cdot)$ have finite support. An immediate question is whether every concept class is linearly separable in this sense. A positive answer would require a construction of the requisite ϕ given $\mathcal{X}$ and $\mathcal{C}$; a negative answer would entail an example of $\mathcal{X}$ and $\mathcal{C}$ for which no such embedding exists.

3 Every concept class is linearly separable

In this section we give an affirmative answer to the question raised in Sec. 2.

Theorem 3.1. *Every countable concept class $\mathcal{C}$ over a countable instance space $\mathcal{X}$ is linearly separable.*

Proof. Let $\mathcal{C}$ be a countable concept class over the countable instance space $\mathcal{X}$. Define two *size* functions on $\mathcal{X}$ and $\mathcal{C}$:

$$|\cdot| : \mathcal{X} \to \mathbb{N}, \qquad \|\cdot\| : \mathcal{C} \to \mathbb{N}$$

with the property that each has finite level sets ($\# f^{-1}(n) < \infty$ for each $n \in \mathbb{N}$); in words, there are at most finitely many elements of a fixed size. Any countable set has such a size function. We will define two auxiliary embeddings, χ and α, and will construct the requisite ϕ as their direct sum. For intuition, it is helpful to keep in mind the dual roles of $\mathcal{X}$ and $\mathcal{C}$. Fix a target $\hat{c} \in \mathcal{C}$.

Define the *embedding by instance* $\chi : \mathcal{X} \to \{0,1\}^{\mathcal{X}}$ by

$$[\chi(x)]_u \;=\; \mathbb{1}_{\{x=u\}}, \qquad u \in \mathcal{X};$$

obviously, $\|\chi(x)\|_0 = 1$ for all $x \in \mathcal{X}$. Define the corresponding hyperplane $w^\chi \in \mathbb{R}^{\mathcal{X}}$ by

$$[w^\chi]_u \;=\; \mathbb{1}_{\{u \in \hat{c}\}} \mathbb{1}_{\{|u| < \|\hat{c}\|\}}, \qquad u \in \mathcal{X};$$

since size functions have finite level sets, we have $\|w^\chi\|_0 < \infty$. Thus,

$$\begin{aligned} \langle w^\chi, \chi(x) \rangle &= \sum_{u \in \mathcal{X}} [w^\chi]_u [\chi(x)]_u \\ &= \sum_{u \in \mathcal{X}} \mathbb{1}_{\{u \in \hat{c}\}} \mathbb{1}_{\{|u| < \|\hat{c}\|\}} \mathbb{1}_{\{x=u\}} \\ &= \mathbb{1}_{\{x \in \hat{c}\}} \mathbb{1}_{\{|x| < \|\hat{c}\|\}}. \end{aligned} \tag{1}$$

Define the *embedding by concept* $\alpha : \mathcal{X} \to \{0,1\}^{\mathcal{C}}$ by

$$[\alpha(x)]_c \;=\; \mathbb{1}_{\{x \in c\}} \mathbb{1}_{\{\|c\| \le |x|\}}, \qquad c \in \mathcal{C};$$

since size functions have finite level sets, we have $\|\alpha(x)\|_0 < \infty$. The corresponding hyperplane $w^\alpha \in \mathbb{R}^{\mathcal{C}}$ is defined by

$$[w^\alpha]_c \;=\; \mathbb{1}_{\{c = \hat{c}\}}, \qquad c \in \mathcal{C}.$$

Now

$$\begin{aligned}\langle w^\alpha, \alpha(x)\rangle &= \sum_{c\in\mathcal{C}} [w^\alpha]_c [\alpha(x)]_c \\ &= \sum_{c\in\mathcal{C}} \mathbb{1}_{\{c=\hat{c}\}} \mathbb{1}_{\{x\in c\}} \mathbb{1}_{\{\|c\|\le|x|\}} \\ &= \mathbb{1}_{\{x\in\hat{c}\}} \mathbb{1}_{\{|x|\ge\|\hat{c}\|\}}. \end{aligned} \tag{2}$$

We define the *canonical* embedding $\phi : \mathcal{X} \to \{0,1\}^{\mathbb{N}}$ as the direct sum of the embeddings by instance and concept:

$$\phi(x) = \chi(x) \oplus \alpha(x);$$

note that

$$\|\phi(x)\|_0 = \|\chi(x)\|_0 + \|\alpha(x)\|_0 < \infty.$$

Similarly, the corresponding hyperplane is the direct sum of the two hyperplanes:

$$w = w^\chi \oplus w^\alpha;$$

again,

$$\|w\|_0 = \|w^\chi\|_0 + \|w^\alpha\|_0 < \infty.$$

Combining (1) and (2), we get

$$\begin{aligned}\langle w, \phi(x)\rangle &= \langle w^\chi, \chi(x)\rangle + \langle w^\alpha, \alpha(x)\rangle \\ &= \mathbb{1}_{\{x\in\hat{c}\}} \mathbb{1}_{\{|x|<\|\hat{c}\|\}} + \mathbb{1}_{\{x\in\hat{c}\}} \mathbb{1}_{\{|x|\ge\|\hat{c}\|\}} \\ &= \mathbb{1}_{\{x\in\hat{c}\}}\end{aligned}$$

which shows that w is indeed a linear separator (with finite support) for $\hat{c}$. □

4 Universal regular kernel

To apply Theorem 3.1 to regular languages (over a fixed alphabet Σ), we observe that the DFAs are a countable concept class $\mathcal{R} = \cup_{n\ge1} \mathrm{DFA}(n)$ over $\mathcal{X} = \Sigma^*$, where $\mathrm{DFA}(n)$ is the set of all DFAs on n states. Denoting by $\|A\|$ the number of states in $A \in \mathcal{R}$, we see that $\|\cdot\|$ is a valid size function on $\mathcal{R}$. A natural size function on Σ^* is string length, denoted by $|\cdot|$. With these two size functions, Theorem 3.1 furnishes an embedding $\phi : \mathcal{R} \to \{0,1\}^{\mathbb{N}}$ that renders all regular languages linearly separable. To get a better feel for this embedding, let us compute its associated kernel

$$\begin{aligned}K(x,y) &= \langle \phi(x), \phi(y)\rangle \\ &= \mathbb{1}_{\{x=y\}} + \sum_{n=1}^{\min\{|x|,|y|\}} K_n(x,y)\end{aligned}$$

where

$$K_n(x,y) = \sum_{A\in\mathrm{DFA}(n)} \mathbb{1}_{\{x\in L(A)\}} \mathbb{1}_{\{y\in L(A)\}}. \tag{3}$$

In other words, $K_n(x, y)$ counts the number of n-state DFAs that accept both x and y. By [5, Theorem 6], an immediate consequence of this construction is that every regular language L can be represented by some *support strings* $\{s_i \in \Sigma^* : 1 \le i \le m\}$ with weights $\alpha \in \mathbb{R}^m$:

$$L = \left\{ x \in \Sigma^* : \sum_{i=1}^{m} \alpha_i K(s_i, x) > 0 \right\}.$$

5 Computing K_n

Since the summation in (3) involves a super-exponential number of terms, brute-force evaluation is out of the question. Though we consider the complexity of K_n to be a likely candidate for #P-complete, we have no proof of this; there is also the hope that the symmetry in the problem will enable a clever efficient computation.

In the meantime, we must resort to a Monte Carlo simulation. For $n > 0$ and $x, y \in \Sigma^*$, define $P_n(x, y)$ to be the fraction of all the DFAs on n states that accept both x and y. Thus, $0 \le P_n(x, y) \le 1$, and computing this quantity is tantamount to computing $K_n(x, y) = P_n(x, y) \, |\mathrm{DFA}(n)|$. Now it is a simple matter to generate n-state DFAs uniformly at random. Let $\{A_i : 1 \le i \le m\}$ be such an independent sample of m-state DFAs, and compute the approximation to $P_n(x, y)$:

$$\hat{P}_n(x, y) = \frac{1}{m} \sum_{i=1}^{m} \mathbb{1}_{\{x \in L(A_i)\}} \mathbb{1}_{\{y \in L(A_i)\}}.$$

Then, by Chernoff's bound, we have

$$\mathbf{P}\left\{ \left| \hat{P}_n(x, y) - P_n(x, y) \right| > \epsilon P_n(x, y) \right\} \le 2 \exp(-\epsilon^2 m P_n(x, y)/3),$$

meaning that with probability at least $1 - 2\exp(-2\epsilon^2 m P_n(x, y))$, we have

$$(1 - \epsilon)\hat{K}(x, y) \le K(x, y) \le (1 + \epsilon)\hat{K}(x, y),$$

where $\hat{K}_n(x, y) = \hat{P}_n(x, y) \, |\mathrm{DFA}(n)|$. Thus, we need

$$m \ge \frac{3 \log(2/\alpha)}{\epsilon^2 P_n(x, y)}$$

sampling steps to have an ϵ-approximation to $K(x, y)$ with probability at least $1 - \alpha$.

It remains to lower-bound $P_n(x, y)$; if it turns out to be exponentially small in automaton size n, the ϵ-approximation will require exponentially many steps. Fortunately, this does not happen:

Theorem 5.1. *For all $n \ge 1$, for all $x, y \in \Sigma^*$, we have*

$$\frac{1}{4} \le P_n(x, y) \le \frac{1}{2}.$$

Proof. The upper bound is simple – it follows from the fact that $K_n(x,x) = \frac{1}{2}\left|\mathrm{DFA}(n)\right|$. Indeed, for any $x \in \Sigma^*$, for every $A^+ \in \mathrm{DFA}(n)$ that accepts x there is exactly one $A^- \in \mathrm{DFA}(n)$ that does not (obtained by changing the state in which A^+ ends up after reading x from accepting to non-accepting). The upper bound follows from the obvious relation $K_n(x,y) \le K_n(x,x)$ for all $x, y \in \Sigma^*$.

To prove the lower bound, take the "worst" case where $x, y \in \Sigma^*$ are such that every $A \in \mathrm{DFA}(n)$ has $\delta(q_0, x) \neq \delta(q_0, y)$. In other words, no automaton ends up in the same state after reading x and as it does after reading y. Since every state is independently chosen to be accepting or not with equal probability, exactly one-fourth of all $A \in \mathrm{DFA}(n)$ will accept both x and y. Clearly, this fraction will be higher if we allow some automata to end up in the same state upon reading x and y. □

This means that if we run the (very simple and efficient) simulation algorithm for $m = 12\epsilon^{-2}\log(2/\alpha)$ steps, we will have an ϵ-approximation to $K_n(x,y)$ with probability at least $1 - \alpha$.

6 Conclusion

Many fascinating questions arise naturally around the kernel K_n that we defined: Is it (or any other universal regular kernel) efficiently computable? How can one efficiently recover the automaton from the hyperplane? Can quantitative margin bounds be obtained (perhaps in terms of automaton size)? These questions hold potential for promising future research.

Acknowledgments

Various key concepts were crystallized during the many valuable discussions with Corinna Cortes and Mehryar Mohri. Thanks also to Jeremiah Blocki, Avrim Blum, Manuel Blum, Daniel Golovin, Nati Linial and Noam Zeilberger for helpful and insightful input.

References

[1] Dana Angluin. On the complexity of minimum inference of regular sets. *Information and Control*, 3(39):337–350, 1978.

[2] Corinna Cortes, Leonid Kontorovich, and Mehryar Mohri. Learning Languages with Rational Kernels. *to appear in COLT*, 2007.

[3] E. Mark Gold. Complexity of automaton identification from given data. *Information and Control*, 3(37):302–420, 1978.

[4] Micheal Kearns and Umesh Vazirani. *An Introduction to Computational Learning Theory.* The MIT Press, 1997.

[5] Leonid Kontorovich, Corinna Cortes, and Mehryar Mohri. Learning Linearly Separable Languages. In *Proceedings of The 17th International Conference on Algorithmic Learning Theory (ALT 2006)*, volume 4264 of *Lecture Notes in Computer Science*, pages 288–303, Barcelona, Spain, October 2006. Springer, Heidelberg, Germany.

[6] Leonard Pitt and Manfred Warmuth. The minimum consistent DFA problem cannot be approximated within any polynomial. *Journal of the Assocation for Computing Machinery*, 40(1):95–142, 1993.

[7] Bernhard Schölkopf and Alex Smola. *Learning with Kernels*. MIT Press, 2002.