arXiv:2302.01097v1 [cs.CL] 2 Feb 2023

基于根加权树自动机的子树核计算新线性时间算法

Ludovic Mignot[a]、Faissal Ouardi[b]、Djelloul Ziadi[a]

[a]鲁昂基础计算机科学研究组，鲁昂诺曼底大学，*76801* 圣艾蒂安迪鲁夫赖 *Cedex*，法国[b] *ANISSE*研究组，计算机科学系，理学院，拉巴特穆罕默德五世大学，摩洛哥

摘要

树核已被提出用于许多领域，例如自然语言应用中的自动学习。本文提出一种基于加权树自动机概念、用于子树核计算的新型线性时间算法。首先，我们引入一类新的加权树自动机，称为根加权树自动机，以及与之相关的形式树级数。随后，我们基于这一类自动机定义子树自动机，用以表示有限树语言的紧凑计算模型。这使我们能够设计一种理论上有保证的线性时间算法，通过加权树自动机交来计算子树核。该算法的核心思想是，以加权树自动机方法所支持的状态等价类计算，取代先前方法中使用的DAG约简和节点排序步骤。我们的方法有三大优势：它是输出敏感的，不受树类型（有序树或无序树）的限制，并且非常适合用于任何基于增量树核的学习方法。最后，我们针对一系列经过调整、适用于深入算法分析的合成树语言数据集，开展了多种对比实验。实验结果表明，所提出的算法优于最先进的方法。

关键词：核方法、结构化数据核、结构化领域学习、树核、加权树自动机、树级数。

1. 引言

核方法已广泛用于扩展许多知名算法的适用范围，例如感知机 [1], 支持向量机 [2], 或主成分分析 [3]。树核是基于机器学习的自然语言处理领域中颇具意义的方法。它们已被用于降低多项自然语言任务的工作量，例如关系抽取 [4], 句法分析重排序 [5], 命名实体识别[6, 7], 语义角色标注 [8],释义检测 [9] 和计算论证 [10]。

电子邮箱地址：通讯作者：f.ouardi@um5r.ac.ma（FaissalOuardi）

在 [11], Haussler提出了一种基于卷积核的框架，用于根据结构化对象各子部分之间的相似性来衡量这些对象的相似性。基于这一思想，人们提出了许多用于树的卷积核，并已成功应用于各种问题。

针对树结构数据，最早提出的核是子树（ST）核 [12] 和子集树（SST）核 [5]。前者定义的特征空间由所有真子树组成，后者则在此基础上纳入子集树。树核研究的发展方向，一是寻找表达能力更强的核函数，二是提升核的计算速度。为处理表达能力与稀疏性问题，研究者将类似 ST 的子结构作为特征：部分树（PT）核 [13]和弹性树核 [14], SubPath [15]。关于树核的更多详情，建议参阅 Da San Martino 的论文 2009 [16]。

本文聚焦于 ST 核的计算。 [12] 中介绍的 ST 核的核心思想是，计算两棵树之间的公共子树数量：它们分别有 m 和 n 个节点。 t_1 和 t_2 可以递归地计算如下：

$$K(t_1, t_2) = \sum_{(n_1, n_2) \in N_{t_1} \times N_{t_2}} \Delta(n_1, n_2) \tag{1}$$

其中， N_{t_1} 和 N_{t_2} 分别是在 t_1 和 t_2中的节点集合， $\Delta(n_1, n_2) = \sum_{i=1}^{|\mathcal{S}|} I_i(n_1)$. 对于某个有限的子树集合 $S = \{s_1, s_2, \cdots\}$， $I_i(n_2)$，而 $I_i(n)$是一个指示函数：若该子树以节点 n 为根，则其值为 1，否则为 0。然后，使用一种字符串匹配算法，将树转换为字符串（详见 [12] ）。该算法的总体计算复杂度为 $O(\max(m, n) \log(\max(m, n)))$，这是该问题的最佳最坏情况时间复杂度。

在 [17], 中，Moschitti 定义了一种用于计算此类树核的算法，该算法可在 $O(m \times n)$时间内计算两棵句法解析树之间的核，其中 m 和 n 是两棵树中的节点数量。因此，Moschitti 通过引入参数 $\sigma \in \{0, 1\}$ 修改了公式（1），该参数可用于子树（$\sigma = 1$）或子集树（$\sigma = 0$）的评估，并针对两棵树 t_1和 t_2 定义如下：如果 n_1 和 n_2 处的产生式不同，则 $\Delta(n_1, n_2) = 0$；如果它们相同且 n_1 和 n_2 是叶节点，则 $\Delta(n_1, n_2) = 1$；最后，如果 n_1 和 n_2 处的产生式相同，且 n_1 和 n_2 不是叶节点，则 $\Delta(n_1, n_2) = \prod_{j=1}^{nc(n_1)} (\sigma + \Delta(C_{n_1}^j, C_{n_2}^j))$，其中 $nc(n_1)$是 n_1 的子节点数量， C_n^j 是节点 n的第j^{th} 个子节点。该算法可通过高效构建节点对集合 $N_p = \{(n_1, n_2) \in N_{t_1} \times N_{t_2} : p(n_1) = p(n_2)\}$来避免在$\Delta(n_1, n_2) = 0$时进行任何评估，其中 $p(n)$返回与 n相关联的产生式规则。该方法需要在预处理步骤中对树的产生式进行排序，然后通过动态规划方法计算 Δ 。该方法的最坏情况时间复杂度为$O(m \times n)$，但在实际应用中能带来相当显著的加速。

2020年，Azais 和 Ingel [18]提出了一种统一框架，基于有向无环图（DAG）约简，从有序树或无序树中计算 ST核，无论其顶点是否带有标签。树森林的 DAG约简被引入为公共子树结构的一种紧凑表示，使得子树核能够快速计算。与那些

基于 Vishwanathan 和 Smola 于 2002年使用的字符串表示方法相比，这种方法的主要优势在于能够快速重复计算 ST核。他们的方法允许实现任意加权函数，而 Aiolli 等人 [19] 提出的 ST核递归计算方法也使用了树数据的 DAG约简，但广泛采用权重的指数形式。他们研究了 DAG 计算各个步骤的理论复杂度，并证明，对于有序树，其复杂度为O(max(m，n))；对于无序树，复杂度为 O(max(m，n) log(max(m，n))（命题 7, [18]）。

在下文中，我们提出一种使用加权树自动机计算 ST核的新方法。我们首先定义一类新的加权树自动机，并称之为根加权树自动机（RWTA）。这类加权树自动机为表示树森林提供了一种高效且最优的新方案，可替代带注释的 DAG 表示。随后，我们证明，利用 RWTA 的一般交运算，并通过状态可达性将其转化为加权树自动机的确定化过程，可以在线性时间内高效计算 ST核。

本文组织如下：第**??** 节概述有秩树和无秩树上的有限树自动机以及正则树语言。接着，在第 2 节中，我们定义一类新的加权树自动机，称为根加权树自动机。然后，在第 3 节中，我们给出子树级数及其自动机的定义。之后，在第 4 节中，我们介绍算法。第一个算法在线性时间内构造 RWTA A_L，作为有限树语言 L 的紧凑表示。第二个算法在时间 $O(|SubTree(X)| + |SubTree(Y)|)$ 内计算 RWTA $A_X \odot A_Y$，以表示加权树自动机 A_X 和 A_Y的 Hadamard 积。第 5 节通过对多种树语言类开展广泛的对比实验，展示我们方法的效率。第 6 节总结全文。

2. 根加权树自动机

设 Σ为一个字母表。 t 上的树由 $t = f(t_1、\ldots\ldots、t_k)$ 归纳定义，其中 k 是任意整数， f是 Σ_k中的任意符号，而 $t_1、\ldots\ldots、t_k$是 Σ上的任意树。我们用 T_Σ 表示 Σ上的树集。 Σ 上的树语言是~~T_Σ的一个~~子集。我们用 $|t|$ 表示树 t 的大小，即其节点数。对于任意树语言 L，定义 $|L| = \sum_{t\in L} |t|$。

集合 S 上的形式树级数 $\mathbb{P}$ 是从 T_Σ 到 S的映射。设 $\mathbb{M} = (M, +)$ 为一个单位元为 0 的幺半群。 $\mathbb{P}$ 的支撑集是集合 Support($\mathbb{P}$) $= \{t \in T_\Sigma \ \mathbb{P}(t) \neq 0\}$。任意形式树级数都等价于一个形式和 $\mathbb{P} = \sum_{t\in T_\Sigma} \mathbb{P}(t)t$。在这种情况下，该形式和被视为满足结合律和交换律。

定义 1. 设 $\mathbb{M} = (M, +)$ 为一个交换幺半群。一个 $\mathbb{M}$-根加权树自动机（$\mathbb{M}$-*RWTA*）是一个四元组 (Σ, Q, ν, δ)，其中：

- $\Sigma = \bigcup_{k\in\mathbb{N}} \Sigma_k$ 一个字母表，
- Q 一个有限状态集，
- ν 一个从 Q到 M 的函数，称为根权重函数，
- δ $Q \times \Sigma_k \times Q^k$ 的一个子集，称为转移集。

在没有歧义时，an $\mathbb{M}$-RWTA 称为 RWTA。

根权重函数 ν 可扩展为 $2^Q \to M$，适用于 Q 的任意子集 S，定义为$\nu(S) = \sum_{s\in S} \nu(s)$。函数 ν等价于 $Q \times M$ 的有限子集，对任意属于 $Q \times M$ 的二元组 $(q,\ m)$，定义为 $(q,\ m) \in \nu \Leftrightarrow \nu(q) = m$。

转移集 δ等价于 $\Sigma_k \times Q^k \to 2^Q$ 中的函数，对任意符号 $f \in \Sigma_k$ 以及任意属于 Q^k 的 k 元组 $(q_1,\ \ldots\ldots,\ q_k)$，定义为

$$q \in \delta(f, q_1, \ldots, q_k) \Leftrightarrow (q, f, q_1, \ldots, q_k) \in \delta.$$

函数 δ 扩展为 $\Sigma_k \times (2^Q)^k \to 2^Q$，定义如下：对任意符号 $f \in \Sigma_k$，以及 Q 的任意 k 元组子集 $(Q_1,\ \ldots\ldots,\ Q_k)$，

$$\delta(f, Q_1, \ldots, Q_k) = \bigcup_{(q_1,\ldots,q_k)\in Q_1\times\cdots\times Q_k} \delta(f, q_1, \ldots, q_k).$$

最后，函数 Δ 将树 $t = f(t_1,\ \ldots\ldots,\ t_k)$ 映射为一个状态集，定义如下：

$$\Delta(t) = \delta(f, \Delta(t_1), \ldots, \Delta(t_k)).$$

当 $t \in \Sigma_0$ 则 $\Delta(t) = \delta(t)$。

与 A 相关联的树的权重为 $\nu(\Delta(t))$。由A 实现的形式树级数是定义在 M 上的形式树级数，记作 $\mathbb{P}_A$，定义为 $\mathbb{P}_A(t) = \nu(\Delta(t))$，其中 $\nu(\emptyset) = 0$，且 0 是 $\mathbb{M}$ 的单位元。Q 中状态 q 的向下语言是集合~~$L_q = \{t \mid q \in \Delta(t)\}$~~。

例 1. 考虑由 $\Sigma_0 = \{a\}$、$\Sigma_1 = \{h\}$和$\Sigma_2 = \{f\}$定义的字母表 Σ。令 $\mathbb{M} = (\mathbb{N}, +)$。由以下内容定义的 *RWTA* $A = (\Sigma, Q, \nu, \delta)$

$$\begin{aligned} Q &= \{1,2,3,4,5\},\\ \nu &= \{(1,0),(2,3),(3,1),(4,2),(5,4)\},\\ \delta &= \{(1,a),(3,a),(2,f,1,3),(4,f,3,3),\\ &\qquad (5,h,2),(5,h,4),(5,h,5)\}, \end{aligned}$$

如图 *1* 所示，并实现树级数：

$$\begin{aligned} \mathbb{P}_A = a + 5f(a,a) + 4h(f(a,a)) + 4h(h(f(a,a))) + \cdots\\ + 4h(h(\ldots h(f(a,a))\ldots)) + \cdots \end{aligned}$$

幺半群 $(M, +)$ 上的 RWTA 可实现的形式树级数类，恰好等于任意半环 $(M, +, \times)$ 上可识别阶跃函数的类 [20,21]。

3. RWTA 与树级数

RWTA 可视为词情形下定义的前缀树。它是一种紧凑结构，可用于表示有限树集。请注意，无环 RWTA 的底层图称为最小有向无环图（DAG），已在基于有向无环图的算法 [22, 16]中引入，作为一种紧凑表示，用于高效计算不同的树核。下文将介绍子树级数及其子树自动机。

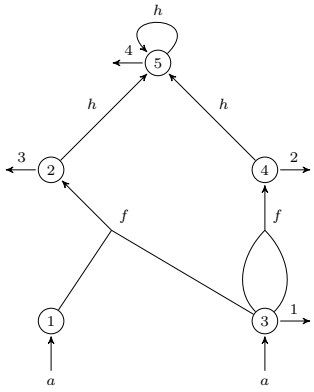


图 1：RWTA $A_{\circ}$

3.1. 子树级数与子树自动机

设 Σ 是一个字母表，$t = f(t_1、\ldots\ldots、t_k)$ 是 T_Σ 中的一棵树。集合 SubTree(t) 是由 SubTree(t) $= \{t\} \cup \bigcup_{1\leq j\leq k}$ SubTree(t_j) 归纳定义的集合。例如，若 $t = f(h(a)\ f(h(a)\ b))$，则 SubTree(t) $= \{a, b, h(a)\ f(h(a)\ b)\ f(h(a)\ f(h(a)\ b))\}_{\circ}$

设 L 是 Σ上的树语言。集合 SubTreeSet(L) 是由 SubTreeSet(L) $= \bigcup_{t\in L}$ SubTree(t) 定义的集合。

形式树级数 SubTreeSeries$_t$ 是由归纳定义的 $\mathbb{N}$ 上的树级数：

SubTreeSeries$_t = t + \sum_{1\leq j\leq k}$ SubTreeSeries$^{t_j}{}_{\circ}$

例2. 设 $t = f(a, g(a))$ 为一棵树。树 t 的子树$_t$ 集合为$\{a, g(a)\ f(a, g(a))\}_{\circ}$

如果 L 是有限的，则有理级数 SubTreeSeries$_L$ 是定义在 $\mathbb{N}$ 上的树级数，其定义为

$$\mathrm{SubTreeSeries}_L = \sum_{t\in L} \mathrm{SubTreeSeries}_t.$$

定义2. 设 Σ 为一个字母表。设 L 为 Σ上的有限树语言。与 L 相关联的子树自动机是 ***RWTA*** $\mathrm{STAut}_L = (\Sigma, Q, Q_m, \nu, \delta)$，其中：

- $Q = \mathrm{SubTreeSet}(L)$,
- $Q_m = L$,
- $\forall t \in Q$, $\nu(t) = \mathrm{SubTreeSeries}_L(t)$,
- $\forall t = f(t_1, \ldots, t_k) \in Q$, $\delta(f, t_1, \ldots, t_k) = \{t\}$.

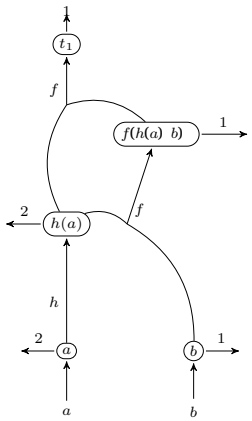


图 2：与树 $f(h(a)\ f(h(a)\ b))$ 相关联的 RWTA。

命题 1. 设 Σ 为一个字母表。设 L为 Σ上的有限树语言。那么，

$$\mathbb{P}_{\mathrm{STAut}_L} = \mathrm{SubTreeSeries}_L.$$

证明。 令 $t = f(t_1, \ldots, t_n)$， $\mathrm{STAut}_t = (\Sigma, Q, Q_m, \nu, \delta)$ 和 $\mathrm{STAut}^{t_i} = (\Sigma, Q_i, Q_{m_i}, \nu_i, \delta_i)$ 对应于
$1 \le i \le k$ $Q = \{t\} \cup \bigcup_{1 \le i \le k} Q_i$ $\delta = \{\ t\ f\ t_1\ \ldots\ t_k\ \} \cup$。注意，根据定义：以及（,,,,）。
$\bigcup_{1 \le i \le k} \delta_i$

因此， $\mathbb{P}_{\mathrm{STAut}_t} = t + \sum_{1 \le i \le k} \mathbb{P}_{A_{t_i}}$ $_t = t +$。
$\sum_{1 \le j \le k}$ $\mathbb{P}_{\mathrm{STAut}_{t_i}} =$
t_j
根据定义，子树级数 子树级数。此外，根据归纳假设，子树级数t_i。因此，有

$$\mathbb{P}_{\mathrm{STAut}_L} = t + \sum_{1 \le j \le k} \mathrm{SubTreeSeries}_{t_j} = \mathrm{SubTreeSeries}_L.$$

□

4. 子树核计算

本节介绍利用树自动机的 Hadamard 积高效计算树核的算法。

4.1. 子树自动机构造

如果自动机 A 与某个 STAut 同构，则称其为 ST 自动机。

本节介绍一种增量算法，用于从有限树集中构造 ST 自动机。

根据构造，ST 自动机是齐次的，即所有进入状态 q 的转移具有相同的标签。因此，我们可以定义函数 h，它将状态 q与其符号 $h(q)$相关联。例如，在图 1 中，我们有 h (1) $= a$、 $h(2) = f$和 $h(4) = f$。由于 $|\delta(f,\ q_1,\ \ldots,\ q_n)| \le 1$， $(q,\ f,\ q_1,\ \ldots,\ q_n) \in \delta \Leftrightarrow \delta(f,\ q_1,\ \ldots,\ q_n) = \{q\}$。因此，我们将 $\delta^{-1}(q)$ 定义为 $f(q_1,\ \ldots,\ q_n)$。

例3。 识别例*2*中的树 $f(a, g(a))$ 的 *RWTA* 的转移为 $\{(1, a)$、$(2, g, 1)$、$(3, f, 1, 2)\}$。函数 δ^{-1} 定义为

$$\delta^{-1}(1) = a, \qquad \delta^{-1}(2) = g(1), \qquad \delta^{-1}(3) = f(1, 2).$$

转移 δ 可以由一个双确定自动机表示（见图 3）。因此，通过 δ 计算 $f(q_1$、.....、$q_n)$ 的像（即 $\delta(f$、q_1、.....、$q_n)$）可在 $O(n)$ 时间内完成，其中 n 是 f的秩。此外，函数 $\delta^{-1}(q)$ 也可以用这个双确定自动机在相同的时间复杂度内计算。

考虑分别与集合 X 和 Y 相关联的两个 ST 自动机 A_X 和 A_Y，它们定义如下:

$$A_X = (\Sigma, Q_X, Q_{m_X}, \nu_X, \delta_X, \delta_X^{\perp}),$$
$$A_Y = (\Sigma, Q_Y, Q_{m_Y}, \nu_Y, \delta_Y, \delta_Y^{\perp}).$$

为了计算 ST 自动机 A_X 和 A_Y之和，我们定义一个从 Q_X 到 Q_Y 的偏函数 ϕ，它将 A_X 和 A_Y 中具有相同向下语言的状态对应起来，即对于 $p \in Q_X$ 和 $p' \in Q_Y$，有 $\phi(p) = p' \Leftrightarrow L_p = L_{p'}$。注意，$\phi$ 是一个定义良好的函数，因为对于一个 ST 自动机，任意不同的状态 p 和q，都有 $L_p \neq L_q$。

算法 1 遍历 A_X 的转移，并在每一步尽可能计算函数 ϕ。因此，如果当前元素 α 是 $f(q_1, \ldots, q_n)$，那么必须定义 $\phi(q_i)$、$1 \leq i \leq n$。为确保这一性质，可以将 A_X 的转移存储在有序列表 OL_δ 中，如下所示：如果 $(q, f, q_1, .., q_n) \in \delta$，那么对所有$1 \leq i \leq n$，$\delta^{-1}(q_i) < \delta^{-1}(q)$。例如，对于例 3 的转移函数，有序列表 OL_δ 为 $[a, g(1)\ f(1, 2)]$。

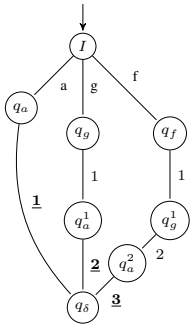


图 3: 与 δ 相关联的双确定自动机。

算法 1 的时间复杂度为 $O(|\delta_X|)$。

命题 2。 设 X 和 Y 是两个有限树集，A_X 和 A_Y 是相应的 *ST* 自动机。那么，可以在 $O(\min(|A_X|, |A_Y|))$ 时间复杂度内计算出 *ST* 自动机 $A_X + A_Y$。

输入： ST 自动机 A_X 和 A_Y
输出： ST 自动机 $A_X + A_Y$
对$\alpha \in \Sigma_0$ **执行**
  如果 $\delta_X(\alpha)$ 和 $\delta_Y(\alpha)$ 存在 **则**
    $\phi(\delta_X(\alpha)) \leftarrow \delta_Y(\alpha)$;
  end
end
$it \leftarrow \mathrm{OL}_{\delta_X}$.获取迭代器();
当it.hasNext() **执行**
  $\alpha \leftarrow$ it.Next(); /*$\alpha = f(q_1, \ldots, q_n)$*/
  如果 $\delta_Y(f(\phi(q_1)\ldots, \phi(q_n)))$ 存在**则**
    $\phi(\delta_X(\alpha)) \leftarrow \delta_Y(f(\phi(q_1)\ldots, \phi(q_n)))$;
    $\nu_Y(\phi(\delta_X(\alpha))) \leftarrow \nu_Y(\delta_Y(\alpha)) + \nu_X(\delta_X(\alpha))$;
  end
  else
    添加一个新状态 q' 到 Q_Y ;
    如果 $\delta_Y(f(\phi(q_1)\ldots, \phi(q_n))) \in Q_{m_X}$ **then**
      将 q' 添加到 Q_{m_Y} ;
    end
    OL_{δ_Y}。添加($f(\phi(q_1)\ldots, \phi(q_n))$ q');
    δ_Y.添加($f(\phi(q_1)\ldots, \phi(q_n))$ q');
    $\nu_Y(q') \leftarrow \nu_X(\delta_X(\alpha))$;
    $\phi(\delta_X(\alpha)) \leftarrow q'$;
  end
end

算法 1： A_X 与 A_Y的并集

注意，可以使用算法 1 计算与树 $t = f(t_1, \ldots, t_n)$ 关联的 ST 自动机。具体来说，首先构造与集合 $\{t_1, \ldots, t_n\}$ 关联的自动机。接下来，若 t_i 对于 $1 \le i \le n$ 在状态 q_{k_i} 被识别，则添加一个新状态 q、一个新转移 $(q, f, q_{k_1}, \ldots, q_{k_n})$，并设置 $Q_m = \{q\}$ 和 $\nu(q) = 1$。

定理 1. 设 Σ 是一个字母表。设 L是 Σ上的有限树语言。那么，与 L 关联的 *ST* 自动机可以在 $O(|L|)$ 时间内计算出来。

4.2. 哈达玛积计算

定义 3. 设 X 和 Y是两个有限树语言。我们定义：子树核(X, Y) $=\sum_{t \in T_\Sigma}$ 子树级数$_X(t)$ $\odot$ 子树级数$_Y(t)$，其中 $\odot$ 是 *Hadamard* 积。

例 4. 设 Σ 是由下式定义的字母表

$$\Sigma_0 = \{a, b\}, \qquad \Sigma_1 = \{h\}, \qquad \Sigma_2 = \{f\}.$$

考虑这些树：$t_1 = f(h(a)\ f(h(a)\ b))$，
$t_2 = f(h(a)\ h(b))$ 且 $t_3 = f(f(b,\ h(b))\ f(h(a)\ h(b)))$。

于是可以证明：

$$\begin{aligned}
\text{SubTreeSeries}_{t_1} &= t_1 + f(h(a), b) + 2h(a) + 2a + b \\
\text{SubTreeSeries}_{t_2} &= t_2 + h(b) + h(a) + a + b \\
\text{SubTreeSeries}_{t_3} &= t_3 + f(b, h(b)) + t_2 \\
&\quad + 2h(b) + h(a) + 3b + a \\
\text{SubTreeSeries}_{\{t_1,t_2\}} &= t_1 + t_2 + f(h(a), b) \\
&\quad + 3h(a) + h(b) + 3a + 2b \\
\text{SubTreeSeries}_{t_3} &= t_3 + f(b, h(b)) + t_2 \\
&\quad + 2h(b) + h(a) + 3b + a \\
\text{SubTreeSeries}_{\{t_1,t_2\}} &\odot \text{SubTreeSeries}_{t_3} \\
&= t_2 + 2h(b) + 3h(a) + 6b + 3a \\
\text{SubTreeKernel}(\{t_1, t_2\}, \{t_3\}) &= 15
\end{aligned}$$

以下命题说明如何根据自动机 A_X 和 A_Y计算 SubTreeKernel(X, Y)。

命题 3. 设 $A_X = (\Sigma, Q_X, Q_{m_X}, \nu_X, \delta_X \uplus \delta_X^{\perp})$ 和 $A_Y = (\Sigma, Q_Y, Q_{m_Y}, \nu_Y, \delta_Y \uplus \delta_Y^{\perp})$ 为两个加权树自动机。*RWTA*$(,,,,)$ $A_X \odot A_Y = (\Sigma, Q_{X\odot Y} \cup \{\perp\}, Q_{m_{X\odot Y}}, \nu_{X\odot Y}, \delta_{X\odot Y} \uplus \delta_{X\odot Y}^{\perp})$ 满足：

- $Q_{X\odot Y} = Q_X \times Q_Y$
- $\delta_{X\odot Y} = \bigcup_{\substack{f\in\Sigma_k, \\ (p,f,p_1,\ldots,p_k)\in\delta_X, \\ (q,f,(q_1,\ldots,q_k))\in\delta_Y}} \{((p,q), f, (p_1,q_1), \ldots, (p_k,q_k))\}$
- $\delta_{X\odot Y}^{\perp} = \{(\perp,\perp)\} \cup \bigcup_{\substack{f\in\Sigma_k, \\ (p,f,p_1,\ldots,p_k)\in\delta_X, \\ (q,f,(q_1,\ldots,q_k))\in\delta_Y}} \{((p,q), f, \perp, \ldots, \perp)\}$
- $\forall (q_1, q_2) \in Q_{X\odot Y}$, $\nu_{X\odot Y}((q_1,q_2)) = \nu_X(q_1) \times \nu_Y(q_2)$.
- $(p,q) \in Q_{m_{X\odot Y}}$ *if* $p \in Q_{m_X} \wedge q \in Q_{m_Y}$,

实现树级数 $\mathbb{P}_{A_X} \odot \mathbb{P}_{A_Y}$。

推论 1. 设 A_X 和 A_Y是两个 α 加权树自动机。则，

$$\text{SubTreeKernel}(X, Y) = \sum_{t\in T_\Sigma} \mathbb{P}_{A_X\odot A_Y}(t).$$

为了高效计算，我们只需计算自动机 $A_X \odot A_Y$的可达部分。该可达部分的大小等于 $|$SubTreeSet(X) $\cap$SubTreeSet(Y)$|$。注意，$|$SubTreeSet(X) $\cap$ SubTreeSet(Y)$| \le |A_X| + |A_Y|$。

以下算法（算法 2）计算两个 ST 自动机的自动机 $A_X \odot A_Y$。

命题 4。 设 A_X 和 A_Y分别是与树集 X 和 Y相关联的 *ST* 自动机。自动机 $A_X \odot A_Y$ 可在时间$O(\min(|\text{SubTreeSet}(X)|, |\text{SubTreeSet}(Y)|))$内计算得出。

输入： ST 自动机 A_X 和 A_Y
输出： $A_X \odot A_Y$
对于$\alpha \in \Sigma_0$ **执行**
  如果 ($\delta_X(\alpha)$ 和 $\delta_Y(\alpha)$) 存在 **则**
    $\phi(\delta_X(\alpha)) \leftarrow \delta_Y(\alpha)$;
  end
end
$it \leftarrow$OL$_{\delta_X}$.获取迭代器();
当it.有下一个()**时执行**
  $\alpha \leftarrow it$.下一个();
  如果 $f(\phi(q_1)\ldots,\phi(q_n))$ 存在 **则**
    令 $\alpha = f(q_1,\ldots,q_n)$;
    令 $p = \delta_Y(f(\phi(q_1)\ldots,\phi(q_n)))$;
    $\phi(\delta_X(\alpha)) \leftarrow p$;
    $\nu_X(\delta_X(\alpha)) = \nu_X(\delta_X(\alpha)) \times \nu_Y(p)$;
  end
  else
    δ_X.移除(α, $\delta_X(\alpha)$);
    it.Remove(α) ;
    $\phi(\delta_X(\alpha)) = \perp$;
  end
end

算法 2： 子树自动机 $A_X \odot A_Y$。

4.3. 核计算

命题 5。 设 X 和 Y 是两个有限树语言。设 Z 是 *ST* 自动机 $A_X \odot A_Y$ 的可达部分。那么，

子树核(X, Y) $= \sum_{q \in Z} \nu(q)$。

由于 $A_X \odot A_Y$ 的可达部分大小有上界 $O(\min(|A_X|, |A_Y|))$，我们可以陈述以下命题。

命题 6. </style}设 和 是两个有限树语言，和 是它们对应的 ST 自动机。那么，子树核 SubTreeKernel(X, Y**) 可以在以下时间内计算**

$O(\min(|A_X|, |A_Y|))$.

最后，我们得到主要结果。

定理 2. 设 X 和 Y 是两个有限树语言。子树核 $\mathrm{SubTreeKernel}(X, Y)$ 可以在 $O(|X| + |Y|)$ 时间和空间复杂度内计算。

这是因为，从集合 X （分别为 Y）中增量构造 A_X （分别为 A_Y）需要 O（$|X|$）（分别为 O（$|Y|$））的时间和空间复杂度，其中 $A_X \leq |X|$ 和 $A_Y \leq |Y|$。

5. 实验与结果

本节通过大量对比实验，从表示规模缩减和执行时间两方面评估基于RWTA 的 ST核计算效率。从算法角度来看，用于此任务的现有真实世界数据集是针对较小树进行学习的标准基准数据集。它们未涵盖深度算法分析所需的多种树特征。

为验证我们方法的效率，我们在如 [22][1] 所述随机生成的合成无序树数据集上进行实验，采用包括字母表大小 $|\mathcal{F}|$、字母表元数的最大值 A以及树深度的最大值 D在内的各种属性组合。对于每种数据集参数组合（$|\mathcal{F}|$、 A、 D），我们均匀且随机地生成一个包含 100 棵树、平均大小为 N 的树集。

下表总结了实验中使用的数据集参数。其中，数据集 DS1（分别为 DS2 和 DS3 ）由五个（分别为四个和七个）树集组成，每个树集的基数均为 100；这些树集通过改变树深度的最大值 D （分别改变字母表元数的最大值 A ，以及同时改变两个参数（A、 D））获得。

表 1：实验中使用的数据集。

	大小 $\|\mathcal{F}\|$	A	D	N
DS1	500	2	5	$[5, 100]$	$[8, 1526]$
DS2	400	2	$[5, 20]$	5	$[8, 1196]$
DS3	700	2	$[2, 15]$	$[5, 100]$	$[5, 478]$

所有算法均使用 C++11 和 Bison++ 解析器实现。编译和汇编使用 gnu-gcc 完成。所有实验均在配备 Intel Core i5–4770K（3.5GHz）中央处理器和 8GB 内存的笔记本电脑上进行。源代码可在此处找到： [23]。

对于数据集参数（$|\mathcal{F}|$， A， D）的每种组合，我们都计算了全部4950种可能树对的ST核。随后，我们计算了所有树对的平均计算时间，以及所构造RWTA的平均状态数。

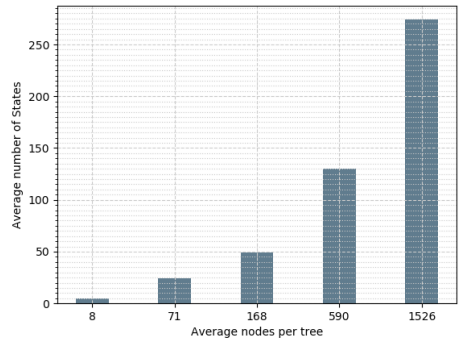
从图4、图5和图6所示的实验结果可以清楚地看出，针对各种树，我们的方法相对于树大小之和具有线性、渐近对数级的复杂度，并且比现有方法更高效。这些结果可以用我们的方法具有输出敏感性来解释。此外，该方法生成了ST核的紧凑表示，可用于增量学习算法。

6. 结论与展望

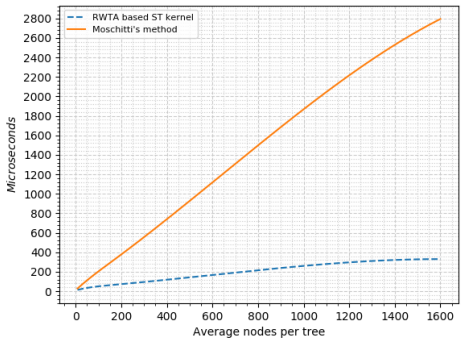
本文定义了新的加权树自动机。给出这些定义后，我们利用这些新结构，高效计算两个有限

[1] http://www.math.unipd.it/∼dasan/pythontreekernels.htm

树语言的子树核。
我们的方法可用于计算其他基于距离的树核，例如 SST核、子路径树核、拓扑树核和 Gappy树核。我们工作的下一步是应用所构造的方法，在基于加权树自动机的统一框架下高效计算这些核。不过，这项应用并非那么直接，因为SST级数似乎无法在具有线性空间复杂度的情况下实现序列化。因此，我们必须寻找其他技术，例如扩展前瞻确定性 [24]。



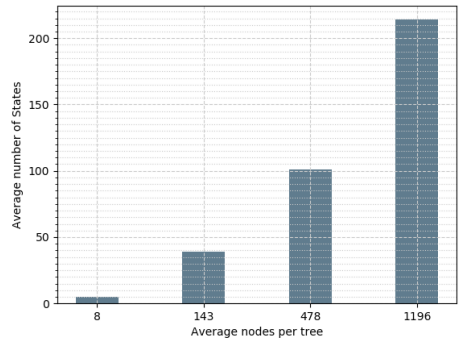
(a)



(b)

图4：(a) 对数据集DS1使用RWTA方法得到的树表示压缩比。(b) 数据集DS1中树深度变化时 ST核的平均计算时间。

(a)



(b)

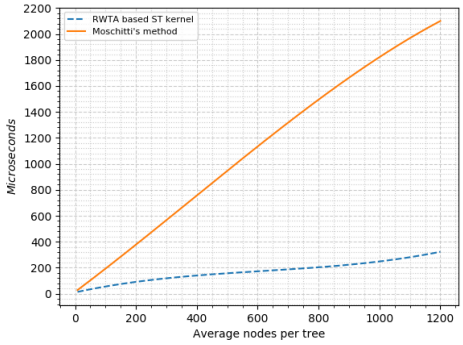
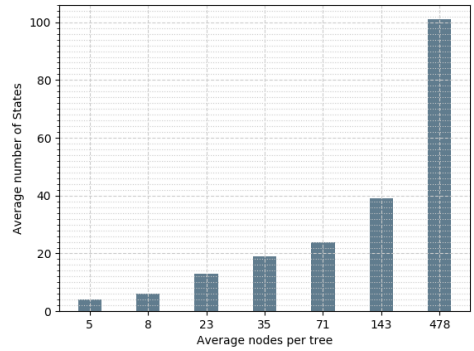
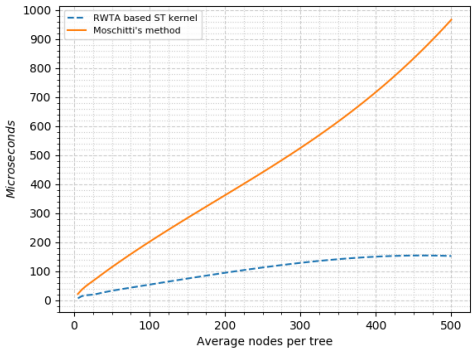


图5：（a）使用RWTA方法对数据集DS2进行树表示的压缩比。（b）数据集DS2中字母表元数变化时ST核的平均计算时间。



(a)



(b)

图6：(a) 使用RWTA方法表示数据集DS3中的树时的压缩比。(b) 数据集DS3中，随着树深度和字母表元数变化，ST核的平均计算时间。

参考文献

[1] M. A. Aizerman、E. M. Braverman、L. I. Rozonoèr，《模式识别中势函数法的理论基础》，Avtomat. i Telemekh. 25 (6) (1964) 917–936。URL http://mi.mathnet.ru/at11677[2] C. Cortes、V. Vapnik，《支持向量网络》，Machine Learning 20 (3) (1995) 273–297。doi:10.1007/BF00994018。URL http://dx.doi.org/10.1007/BF00994018[3] D. Zelenko、C. Aone、A. Richardella，《用于关系抽取的核方法》，Journal of Machine Learning Research 3 (2003)1083–1106。URL http://www.jmlr.org/papers/v3/zelenko03a.html[4] B. Schölkopf、A. J. Smola、K. Müller，《核主成分分析》，载于：Artificial Neural Networks - ICANN '97，第 7 届国际会议，

洛桑，瑞士，1997 年 10 月 8–10 日，会议论文集，1997 年，第 583–588 页。doi:10.1007/BFb0020217。URL http://dx.doi.org/10.1007/BFb0020217

[5] M. Collins, N. Duffy，《用于句法分析和标注的新排序算法：离散结构上的核与投票感知机》，载于：第40届计算语言学协会年会论文集，2002年7月6日至12日，美国宾夕法尼亚州费城，2002年，页码263–270。网址：http://www.aclweb.org/anthology/P02-1034.pdf[6] A. Culotta, J. S. Sorensen，《用于关系抽取的依存树核》，载于：第42届计算语言学协会年会论文集，2004年7月21日至26日，西班牙巴塞罗那，2004年，页码423–429。网址：http://acl.ldc.upenn.edu/acl2004/main/pdf/244_pdf_2-col.pdf[7] C. M. Cumby, D. Roth，《关系学习的核方法》，载于：Machine Learning，第二十届国际会议（ICML 2003）论文集，2003年8月21日至24日，美国华盛顿特区，2003年，页码107–114。网址：http://www.aaai.org/Library/ICML/2003/icml03-017. php[8] A. Moschitti，《浅层统计句法分析的卷积核研究》，载于：第42届计算语言学协会年会论文集，2004年7月21日至26日，西班牙巴塞罗那，2004年，页码335–342。网址：http://acl.ldc.upenn.edu/acl2004/main/pdf/388_pdf_2-col.pdf[9] S. Filice, G. Da San Martino, A. Moschitti，《用于学习文本对之间关系的结构表示》，载于：第53届计算语言学协会年会暨第7届自然语言处理国际联合会议论文集（第1卷：长篇论文），计算语言学协会，中国北京，2015年，页码1003–1013。doi:10.3115/v1/P15-1097。网址：https://www.aclweb.org/anthology/P15-1097[10] H. Wachsmuth, N. Naderi, Y. Hou, Y. Bilu, V. Prabhakaran, T. A. Thijm, G. Hirst, B. Stein，《自然语言中的计算论证质量评估》，载于：第15届欧洲计算语言学协会分会会议论文集：第1卷，长篇论文，计算语言学协会，西班牙瓦伦西亚，2017年，页码176–187。网址：https://www.aclweb.org/anthology/E17-1017[11] D. Haussler，《离散结构上的卷积核》，技术报告，计算机科学系，加利福尼亚大学……（1999年）。[12] S. Vishwanathan, A. J. Smola.，《字符串和树上的快速核》，《神经信息处理系统进展》，（2002）14。

[13] A. Moschitti, 《用于依存句法树和成分句法树的高效卷积核》，载于：ECML，《计算机科学讲义》第4212卷，施普林格，2006年，页码318–329。[14] H. Kashima, T. Koyanagi, 《半结构化数据的核》，载于：ICML，Morgan Kaufmann，2002年，页码291–298。[15] D. Kimura, T. Kuboyama, T. Shibuya, H. Kashima, 《有根无序树的子路径核》，载于：PAKDD (1)，《计算机科学讲义》第6634卷，施普林格，2011年，页码62–74。[16] D. S. M. G., 《树结构数据的核方法》，博士论文（2009年）。[17] A. Moschitti, 《让树核适用于自然语言学习》，载于：EACL 2006，欧洲计算语言学协会分会第11届会议论文集，2006年4月3日至7日，意大利特伦托，2006年，页码113–120。网址：http://acl.ldc.upenn.edu/E/E06/E06-1015.pdf[18] R. Azaïs, F. Ingels, 《子树核中的权重函数至关重要》，Journal of Machine Learning Research 21 (2020年) 1–36。网址：https://jmlr.org/papers/v21/19-290.html[19] F. Aiolli, G. Da San Martino, A. Sperduti, A. Moschitti, 《面向树的高效基于核的学习》，载于：2007年IEEE计算智能与数据挖掘研讨会，2007年，页码308–315。doi:10.1109/CIDM.2007.368889。[20] M. Droste, P. Gastin, 《加权自动机与加权逻辑》，Theor. Comput. Sci. 380 (1-2) (2007年) 69–86。doi:10.1016/j.tcs.2007.02.055。网址：http://dx.doi.org/10.1016/j.tcs.2007.02.055[21] M. Droste, H. Vogler, 《加权树自动机与加权逻辑》，Theor. Com- put. Sci. 366 (3) (2006年) 228–247。doi:10.1016/j.tcs.2006.08.025。网址：http://dx.doi.org/10.1016/j.tcs.2006.08.025[22] F. Aiolli, G. D. S. Martino, A. Sperduti, 《一种高效的基于拓扑距离的树核》，IEEE神经网络与学习系统汇刊 26 (2015年) 1115–1120。[23] F. Ouardi, 《使用RWTA进行子树核计算》，https://github.com/ouardifaissal/Subtree-kernel-computation-using-RWTA（2021年）。[24] Y. Han, D. Wood, 《1-确定正则语言的推广》，Information and Computation 206 (9-10) (2008年) 1117–1125。doi:10.1016/j.ic.2008.03.013。