Learning Concepts Described By Weight Aggregation Logic

Steffen van Bergerem
RWTH Aachen University, Germany
vanbergerem@informatik.rwth-aachen.de

Nicole Schweikardt
Humboldt-Universität zu Berlin, Germany
schweikn@informatik.hu-berlin.de

Abstract

We consider weighted structures, which extend ordinary relational structures by assigning weights, i.e. elements from a particular group or ring, to tuples present in the structure. We introduce an extension of first-order logic that allows to aggregate weights of tuples, compare such aggregates, and use them to build more complex formulas. We provide locality properties of fragments of this logic including Feferman-Vaught decompositions and a Gaifman normal form for a fragment called FOW_1, as well as a localisation theorem for a larger fragment called FOWA_1. This fragment can express concepts from various machine learning scenarios. Using the locality properties, we show that concepts definable in FOWA_1 over a weighted background structure of at most polylogarithmic degree are agnostically PAC-learnable in polylogarithmic time after pseudo-linear time preprocessing.

2012 ACM Subject Classification Theory of computation → Logic; Theory of computation → Complexity theory and logic; Computing methodologies → Logical and relational learning; Computing methodologies → Supervised learning

Keywords and phrases first-order definable concept learning, agnostic probably approximately correct learning, classification problems, locality, Feferman-Vaught decomposition, Gaifman normal form, first-order logic with counting, weight aggregation logic

Digital Object Identifier 10.4230/LIPIcs.CSL.2021.10

Related Version A full version of the paper is available at [23], `https://arxiv.org/abs/2009.10574`.

Funding The second author has been partially supported by the ANR project EQUUS ANR-19-CE48-0019; funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – project number 431183758 (gefördert durch die Deutsche Forschungsgemeinschaft (DFG) – Projektnummer 431183758).

Acknowledgements We thank Martin Grohe and Sandra Kiefer for helpful discussions on the subject.

1 Introduction

In this paper, we study Boolean classification problems. The elements that are to be classified come from a set $\mathcal{X}$, the *instance space*. A *classifier* on $\mathcal{X}$ is a function $c\colon \mathcal{X} \to \{0,1\}$. Given a *training sequence* T of labelled examples $(x_i, b_i) \in \mathcal{X} \times \{0,1\}$, we want to find a classifier, called a *hypothesis*, that can be used to predict the label of elements from $\mathcal{X}$ not given in T. We consider the following well-known frameworks for this setting from computational learning theory.

In Angluin's model of *exact learning* [1], the examples are assumed to be generated using an unknown classifier, the *target concept*, from a known *concept class*. The task is to find a hypothesis that is consistent with the training sequence T, i.e. a function $h\colon \mathcal{X} \to \{0,1\}$ such that $h(x_i) = b_i$ for all i. In Haussler's model of *agnostic probably approximately correct (PAC) learning* [11], a generalisation of Valiant's *PAC learning* model [21], an (unknown) probability distribution $\mathcal{D}$ on $\mathcal{X} \times \{0, 1\}$ is assumed and training examples are drawn independently from this distribution. The goal is to find a hypothesis that generalises well, i.e. one is interested in algorithms that return with high probability a hypothesis with a small expected error on new instances drawn from the same distribution. For more background on PAC learning, we refer to [12, 19]. We study learning problems in the framework that was introduced by Grohe and Turán [9] and further studied in [3, 6, 7, 22]. There, the instance space $\mathcal{X}$ is a set of tuples from a background structure and classifiers are described using parametric models based on logics.

Our Contribution. We introduce a new logic for describing such classifiers, namely *first-order logic with weight aggregation* (FOWA). It operates on *weighted structures*, which extend ordinary relational structures by assigning weights, i.e. elements from a particular abelian group or ring, to tuples present in the structure. Such weighted structures were recently considered by Toruńczyk [20], who studied the complexity of query evaluation problems for the related logic FO[$\mathbb{C}$] and its fragment $\text{FO}_{\text{G}}[\mathbb{C}]$. Our logic FOWA, however, is closer to the syntax and semantics of the first-order logic with counting quantifiers FOC considered in [13]. This connection enables us to achieve locality results for the fragments FOW_1 and FOWA_1 of FOWA similar to those obtained in [14, 8]. Specifically, we achieve Feferman-Vaught decompositions and a Gaifman normal form for FOW_1 as well as a localisation theorem for the more expressive logic FOWA_1. We provide examples illustrating that FOWA_1 can express concepts relevant for various machine learning scenarios. Using the locality properties, we show that concepts definable in FOWA_1 over a weighted background structure of at most polylogarithmic degree are agnostically PAC-learnable in polylogarithmic time after pseudo-linear time preprocessing. This generalises the results that Grohe and Ritzert [7] obtained for first-order logic to the substantially more expressive logic FOWA_1.

The main drawback of the existing logic-based learning results is that they deal with structures and logics that are too weak for describing meaningful classifiers for real-world machine learning problems. In machine learning, input data is often given via numerical values which are contained in or extracted from a more complex structure, such as a relational database (cf., [5, 10, 17, 18]). Hence, to combine these two types of information, we are interested in hybrid structures, which extend relational ones by numerical values. Just as in commonly used relational database systems, to utilise the power of such hybrid structures, the classifiers should be allowed to use different methods to aggregate the numerical values. Our main contribution is the design of a logic that is capable of expressing meaningful machine learning problems and, at the same time, well-behaved enough to have similar locality properties as first-order logic, which enable us to learn the concepts in sublinear time.

Outline. This paper is structured as follows. Section 2 fixes basic notation. Section 3 introduces the logic FOWA and its fragments FOW_1 and FOWA_1, provides examples, and discusses enrichments of the logic with syntactic sugar in order to make it more user-friendly (i.e. easier to parse or construct formulas) without increasing its expressive power. Section 4 provides locality results for the fragments FOW_1 and FOWA_1 that are similar in spirit to the known locality results for first-order logic and the counting logic FOC_1. Section 5 is devoted to our results on agnostic PAC learning. Section 6 combines the results from the previous sections to obtain our main learning theorem for FOWA_1, and concludes the paper with an application scenario and directions for future work. Due to space restrictions, we omit some proofs and proof details in this article; all these details can be found in the preliminary full version of this paper [23].

2 Preliminaries

Standard Notation. We write $\mathbb{R}$, $\mathbb{Q}$, $\mathbb{Z}$, $\mathbb{N}$, and $\mathbb{N}_{\geqslant 1}$ for the sets of reals, rationals, integers, non-negative integers, and positive integers, respectively. For all $m, n \in \mathbb{N}$, we write $[m, n]$ for the set $\{k \in \mathbb{N} : m \leqslant k \leqslant n\}$, and we let $[m] := [1, m]$. For a k-tuple $\bar{x} = (x_1, \dots, x_k)$, we write $|\bar{x}|$ to denote its *arity* k. By $()$, we denote the empty tuple, i.e. the tuple of arity 0. All graphs are assumed to be undirected. For a graph G, we write $V(G)$ and $E(G)$ to denote its vertex set and edge set, respectively. For $V' \subseteq V(G)$, we write $G[V']$ to denote the subgraph of G induced on V'. We assume familiarity with standard definitions concerning groups and rings (cf., [23]). When referring to an abelian group (or ring), we will usually write $(S, +_S)$ (or $(S, +_S, \cdot_S)$), we denote the neutral element of the group by 0_S, and $-a$ denotes the inverse of an element a in $(S, +_S)$ (and we denote the neutral element of the ring for $(S, \cdot_S)$ by 1_S).

Signatures, Structures, and Neighbourhoods. A *signature* σ is a finite set of relation symbols. Associated with every $R \in \sigma$ is an arity $\operatorname{ar}(R) \in \mathbb{N}$. A *$\sigma$-structure* $\mathcal{A}$ consists of a finite non-empty set A called the *universe* of $\mathcal{A}$ (sometimes denoted $U(\mathcal{A})$), and for each $R \in \sigma$ a relation $R^{\mathcal{A}} \subseteq A^{\operatorname{ar}(R)}$. The *size* of $\mathcal{A}$ is $|\mathcal{A}| := |A|$. Note that, according to these definitions, the universe $A = U(\mathcal{A})$ of each considered structure $\mathcal{A}$ is finite, and all considered signatures σ are *relational* (i.e. they do not contain any constants or function symbols), and may contain relation symbols of arity 0 (the only 0-ary relations over a set A are $\emptyset$ and $\{()\}$).

Let σ' be a signature with $\sigma' \supseteq \sigma$. A *σ'-expansion* of a σ-structure $\mathcal{A}$ is a σ'-structure $\mathcal{B}$ with universe B such that $B = A$ and $R^{\mathcal{B}} = R^{\mathcal{A}}$ for every $R \in \sigma$. If $\mathcal{B}$ is a σ'-expansion of $\mathcal{A}$, then $\mathcal{A}$ is called the *σ-reduct* of $\mathcal{B}$. A *substructure* of a σ-structure $\mathcal{A}$ is a σ-structure $\mathcal{B}$ with a universe $B \subseteq A$ and $R^{\mathcal{B}} \subseteq R^{\mathcal{A}}$ for all $R \in \sigma$. For a σ-structure $\mathcal{A}$ and a non-empty set $B \subseteq A$, we write $\mathcal{A}[B]$ to denote the *induced substructure* of $\mathcal{A}$ on B, i.e. the σ-structure with universe B and $R^{\mathcal{A}[B]} = R^{\mathcal{A}} \cap B^{\operatorname{ar}(R)}$ for every $R \in \sigma$.

The *Gaifman graph* $G_{\mathcal{A}}$ of a σ-structure $\mathcal{A}$ is the graph with vertex set A and an edge between two distinct vertices $a, b \in A$ iff there exists $R \in \sigma$ and a tuple $(a_1, \dots, a_{\operatorname{ar}(R)}) \in R^{\mathcal{A}}$ such that $a, b \in \{a_1, \dots, a_{\operatorname{ar}(R)}\}$. The structure $\mathcal{A}$ is *connected* if $G_{\mathcal{A}}$ is connected; the *connected components* of $\mathcal{A}$ are the connected components of $G_{\mathcal{A}}$. The *degree* of $\mathcal{A}$ is the degree of $G_{\mathcal{A}}$, i.e. the maximum number of neighbours of a vertex of $G_{\mathcal{A}}$. The *distance* $\operatorname{dist}^{\mathcal{A}}(a, b)$ between two elements $a, b \in A$ is the minimal number of edges of a path from a to b in $G_{\mathcal{A}}$; if no such path exists, we set $\operatorname{dist}^{\mathcal{A}}(a, b) := \infty$. For a tuple $\bar{a} = (a_1, \dots, a_k) \in A^k$ and an element $b \in A$, we let $\operatorname{dist}^{\mathcal{A}}(\bar{a}, b) := \min_{i \in [k]} \operatorname{dist}(a_i, b)$, and for a tuple $\bar{b} = (b_1, \dots, b_\ell)$, we let $\operatorname{dist}(\bar{a}, \bar{b}) := \min_{j \in [\ell]} \operatorname{dist}(\bar{a}, b_j)$.

For every $r \geqslant 0$, the *r-ball of $\bar{a}$ in $\mathcal{A}$* is the set $N_r^{\mathcal{A}}(\bar{a}) = \{b \in A : \operatorname{dist}^{\mathcal{A}}(\bar{a}, b) \leqslant r\}$. The *$r$-neighbourhood of $\bar{a}$ in $\mathcal{A}$* is the structure $\mathcal{N}_r^{\mathcal{A}}(\bar{a}) := \mathcal{A}[N_r^{\mathcal{A}}(\bar{a})]$.

3 Weight Aggregation Logic

This section introduces our new logic, which we call *first-order logic with weight aggregation*. It is inspired by the counting logic FOC and its fragment FOC_1, as introduced in [13, 8], as well as the logic $\mathrm{FO}[\mathbb{C}]$ and its fragment $\mathrm{FO}_{\mathrm{G}}[\mathbb{C}]$, which were recently introduced by Toruńczyk in [20]. Similarly as in [20], we consider *weighted structures*, which extend ordinary relational structures by assigning a weight, i.e. an element of a particular group or ring, to tuples present in the structure. The syntax and semantics of our logic, however, are closer in spirit to the syntax and semantics of the logic FOC_1, since this will enable us to achieve locality results similar to those obtained in [14, 8].

Weighted Structures. Let σ be a signature. Let $\mathbb{S}$ be a collection of rings and/or abelian groups. Let $\mathbf{W}$ be a finite set of *weight symbols*, such that each $\mathtt{w} \in \mathbf{W}$ has an associated *arity* $\mathrm{ar}(\mathtt{w}) \in \mathbb{N}_{\geqslant 1}$ and a *type* $\mathrm{type}(\mathtt{w}) \in \mathbb{S}$. A $(\sigma, \mathbf{W})$*-structure* is a σ-structure $\mathcal{A}$ that is enriched, for every $\mathtt{w} \in \mathbf{W}$, by an interpretation $\mathtt{w}^{\mathcal{A}} \colon A^{\mathrm{ar}(\mathtt{w})} \to \mathrm{type}(\mathtt{w})$, which satisfies the following *locality condition*: if $\mathtt{w}^{\mathcal{A}}(a_1, \dots, a_k) \neq 0_S$ for $S := \mathrm{type}(\mathtt{w})$, $k := \mathrm{ar}(\mathtt{w})$ and $(a_1, \dots, a_k) \in A^k$, then $k = 1$ or $a_1 = \cdots = a_k$ or there exists an $R \in \sigma$ and a tuple $(b_1, \dots, b_{\mathrm{ar}(R)}) \in R^{\mathcal{A}}$ such that $\{a_1, \dots, a_k\} \subseteq \{b_1, \dots, b_{\mathrm{ar}(R)}\}$. All notions that were introduced in Section 2 for σ-structures carry over to $(\sigma, \mathbf{W})$-structures in the obvious way. Specifically, if $\mathcal{A}$ is a $(\sigma, \mathbf{W})$-structure and σ' is a signature with $\sigma' \supseteq \sigma$, then a σ'*-expansion of* $\mathcal{A}$ is a $(\sigma', \mathbf{W})$-structure $\mathcal{B}$ with $B = A$, $R^{\mathcal{B}} = R^{\mathcal{A}}$ for all $R \in \sigma$, and $\mathtt{w}^{\mathcal{B}} = \mathtt{w}^{\mathcal{A}}$ for all $\mathtt{w} \in \mathbf{W}$.

We will use the following as running examples throughout this section.

▶ **Example 3.1.**

(a) Consider an online marketplace that allows retailers to sell their products to consumers. The database of the marketplace contains a table with transactions, and each entry consists of an identifier, a customer, a product, a retailer, the price per item, and the number of items sold. We can describe the database of the marketplace as a weighted structure as follows. Let $(\mathbb{Q}, +, \cdot)$ be the field of rationals, let $\mathbf{W}$ contain two unary weight symbols $\mathtt{price}$ and $\mathtt{quantity}$ of type $(\mathbb{Q}, +, \cdot)$, let $\sigma = \{T\}$, and let $\mathcal{A}$ be a $(\sigma, \mathbf{W})$-structure such that the universe A contains the identifiers for the transactions, customers, products, and retailers. For every transaction, let $T^{\mathcal{A}}$ contain the 4-tuple (i, c, p, r) consisting of the identifier for the transaction, the customer, the product, and the retailer. For every transaction identifier i, let $\mathtt{price}^{\mathcal{A}}(i)$ be the price per item in the transaction and let $\mathtt{quantity}^{\mathcal{A}}(i)$ be the number of items sold; for every other identifier a in A, let $\mathtt{price}^{\mathcal{A}}(a) = \mathtt{quantity}^{\mathcal{A}}(a) = 0$.

(b) In a recent survey [17], Pan and Ding describe different approaches to represent social media users via embeddings into a low-dimensional vector space, where the embeddings are based on the users' social media posts[1]. We represent the available data by a weighted structure $\mathcal{A}$ as follows. Consider the group $(\mathbb{R}^k, +)$, where $\mathbb{R}^k$ is the set of k-dimensional real vectors and $+$ is the usual vector addition, and let $\mathbf{W}$ contain a unary weight symbol $\mathtt{embedding}$ of type $(\mathbb{R}^k, +)$. Let $\sigma = \{F\}$ and let $\mathcal{A}$ be a $(\sigma, \mathbf{W})$-structure such that the universe A consists of the users of a social network. Let $F^{\mathcal{A}}$ contain all pairs of users (a, b) such that a is a follower of b. For every user $a \in A$, let $\mathtt{embedding}^{\mathcal{A}}(a)$ be a k-dimensional vector representing a's social media posts.

(c) Consider vertex-coloured edge-weighted graphs, where R, B, G are unary relations of red, blue, and green vertices, E is a binary relation of edges, and where every edge (a, b) has an associated *weight* that is a k-dimensional vector of reals (for some fixed number k). Such graphs can be viewed as $(\sigma, \mathbf{W})$-structures $\mathcal{A}$, where $\sigma = \{E, R, B, G\}$, $\mathbf{W}$ contains a binary weight symbol $\mathtt{w}$ of type $(\mathbb{R}^k, +)$ and $\mathtt{w}^{\mathcal{A}}(a, b) \in \mathbb{R}^k$ for all edges $(a, b) \in E^{\mathcal{A}}$.

Fix a countably infinite set vars of *variables.* A $(\sigma, \mathbf{W})$*-interpretation* $\mathcal{I} = (\mathcal{A}, \beta)$ consists of a $(\sigma, \mathbf{W})$-structure $\mathcal{A}$ and an *assignment* $\beta \colon \mathsf{vars} \to A$. For $k \in \mathbb{N}_{\geqslant 1}$, elements $a_1, \dots, a_k \in A$, and k distinct variables $y_1, \dots, y_k$, we write $\mathcal{I}\frac{a_1, \dots, a_k}{y_1, \dots, y_k}$ for the interpretation $(\mathcal{A}, \beta\frac{a_1, \dots, a_k}{y_1, \dots, y_k})$, where $\beta\frac{a_1, \dots, a_k}{y_1, \dots, y_k}$ is the assignment β' with $\beta'(y_i) = a_i$ for every $i \in [k]$ and $\beta'(z) = \beta(z)$ for all $z \in \mathsf{vars} \setminus \{y_1, \dots, y_k\}$.

[1] Among other applications, such embeddings might be used to predict a user's personality or political leaning.

The Weight Aggregation Logic FOWA and its Restrictions FOWA_1 and FOW_1. Let σ be a signature, $\mathbb{S}$ a collection of rings and/or abelian groups, and $\mathbf{W}$ a finite set of weight symbols. An $\mathbb{S}$-*predicate collection* is a 4-tuple $(\mathbb{P}, \text{ar}, \text{type}, [\![\cdot]\!])$ where $\mathbb{P}$ is a countable set of *predicate names* and, to each $\mathsf{P} \in \mathbb{P}$, ar assigns an *arity* $\text{ar}(\mathsf{P}) \in \mathbb{N}_{\geqslant 1}$, type assigns a *type* $\text{type}(\mathsf{P}) \in \mathbb{S}^{\text{ar}(\mathsf{P})}$, and $[\![\cdot]\!]$ assigns a *semantics* $[\![\mathsf{P}]\!] \subseteq \text{type}(\mathsf{P})$. For the remainder of this section, fix an $\mathbb{S}$-predicate collection $(\mathbb{P}, \text{ar}, \text{type}, [\![\cdot]\!])$.

For every $S \in \mathbb{S}$ that is not a ring but just an abelian group, a $\mathbf{W}$-*product of type* S is either an element $s \in S$ or an expression of the form $\mathtt{w}(y_1, \ldots, y_k)$ where $\mathtt{w} \in \mathbf{W}$ is of type S, $k = \text{ar}(\mathtt{w})$, and $y_1, \ldots, y_k$ are k pairwise distinct variables in vars. For every ring $S \in \mathbb{S}$, a $\mathbf{W}$-*product of type* S is an expression of the form $t_1 \cdot \cdots \cdot t_\ell$ where $\ell \in \mathbb{N}_{\geqslant 1}$ and for each $i \in [\ell]$ either $t_i \in S$ or there exists a $\mathtt{w} \in \mathbf{W}$ with $\text{type}(\mathtt{w}) = S$ and there exist $k := \text{ar}(\mathtt{w})$ pairwise distinct variables $y_1, \ldots, y_k$ in vars such that $t_i = \mathtt{w}(y_1, \ldots, y_k)$. By $\text{vars}(p)$ we denote the set of all variables that occur in a $\mathbf{W}$-product p.

▶ **Example 3.2.** Recall Example 3.1(a)–(c), and let x and y be variables. Examples of $\mathbf{W}$-products are $\mathtt{price}(x) \cdot \mathtt{quantity}(x)$, $\mathtt{embedding}(x)$, and $\mathtt{w}(x, y)$. Below, in Definition 3.3, we will provide the formal definition of a logic (including notions of *formulas* and so-called $\mathbb{S}$-*terms*) which is capable of expressing the following statements.

(a) Given a first-order formula $\varphi_{\text{group}}(p)$ that defines products of a certain product group based on the structure of their transactions, we can describe the amount of money a consumer c paid on the specified product group via the $\mathbb{S}$-*term*

$$t_{\text{spending}}(c) := \sum \mathtt{price}(i) \cdot \mathtt{quantity}(i)\ .\ \exists p\, \exists r\, \big(\varphi_{\text{group}}(p) \wedge T(i, c, p, r)\big).$$

This term associates with every consumer c the sum of the product of $\mathtt{price}(i)$ and $\mathtt{quantity}(i)$ for all transaction identifiers i for which there exists a product p and a retailer r such that the tuple (i, c, p, r) belongs to the transaction table and $\varphi_{\text{group}}(p)$ holds. The $\mathbb{S}$-term

$$t_{\text{sales}} := \sum \mathtt{price}(i) \cdot \mathtt{quantity}(i)\ .\ \exists c\, \exists p\, \exists r\, \big(\varphi_{\text{group}}(p) \wedge T(i, c, p, r)\big)$$

specifies the amount *all* customers have paid on products from the product group.
We might want to select the “heavy hitters”, i.e. all customers c for whom $t_{\text{spending}}(c) > 0.01 \cdot t_{\text{sales}}$ holds. In our logic, this is expressed by the formula

$$\mathsf{P}_{>}(t_{\text{spending}}(c), 0.01 \cdot t_{\text{sales}})\,, \quad \text{where}$$

$\mathsf{P}_{>}$ is a predicate name of type $(\mathbb{Q}, +, \cdot) \times (\mathbb{Q}, +, \cdot)$ with $[\![\mathsf{P}_{>}]\!] = \{(r, s) \in \mathbb{Q}^2 : r > s\}$.

(b) For vectors $u, v \in \mathbb{R}^k$, let $d(u, v)$ denote the Euclidean distance between u and v. We might want to use a formula $\varphi_{\text{similar}}(x, y)$ expressing that the two k-dimensional vectors associated with persons x and y have Euclidean distance at most 1. To express this in our logic, we can add the rational field $(\mathbb{Q}, +, \cdot)$ to the collection $\mathbb{S}$ and use a predicate name P_{ED} of arity 3 and type $(\mathbb{R}^k, +) \times (\mathbb{R}^k, +) \times (\mathbb{Q}, +, \cdot)$ with $[\![\mathsf{P}_{\text{ED}}]\!] = \{(u, v, q) \in \mathbb{R}^k \times \mathbb{R}^k \times \mathbb{Q} : d(u, v) \leqslant q\}$. Then,

$$\varphi_{\text{similar}}(x, y) := \mathsf{P}_{\text{ED}}(\mathtt{embedding}(x), \mathtt{embedding}(y), 1)$$

is a formula with the desired meaning.

(c) For each vertex x, the sum of the weights of edges between x and its blue neighbours is specified by the $\mathbb{S}$-term $t_B(x) := \sum \mathtt{w}(x', y).(x'{=}x \wedge E(x', y) \wedge B(y))$.

We have designed the definition of the syntax of our logic in a way particularly suitable for formulating and proving the locality results that are crucial for obtaining our learning results. To obtain a more user-friendly syntax, i.e. which allows to read and construct formulas in a more intuitive way, one could of course introduce syntactic sugar that allows to explicitly write statements of the form

- $t_{\text{spending}}(c) > 0.01 \cdot t_{\text{sales}}$ instead of $\mathsf{P}_{>}(t_{\text{spending}}(c), 0.01 \cdot t_{\text{sales}})$
- $d(\text{embedding}(x), \text{embedding}(y)) \leqslant 1$ instead of $\mathsf{P}_{\text{ED}}(\text{embedding}(x), \text{embedding}(y), 1)$
- $\sum\limits_{y} \mathtt{w}(x,y).(E(x,y) \wedge B(y))$ instead of $\sum \mathtt{w}(x',y).(x'{=}x \wedge E(x',y) \wedge B(y))$.

We now define the precise syntax and semantics of our weight aggregation logic.

▶ **Definition 3.3** ($\mathsf{FOWA}(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$). *For* $\text{FOWA}(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$, *the set of* formulas *and* $\mathbb{S}$-terms *is built according to the following rules:*

(1) $x_1{=}x_2$ *and* $R(x_1,\dots,x_{\text{ar}(R)})$ *are* formulas,
where $R \in \sigma$ *and* $x_1, x_2, \dots, x_{\text{ar}(R)}$ *are variables*[2].

(2) *If* $\mathtt{w} \in \mathbf{W}$, $S = \text{type}(\mathtt{w})$, $s \in S$, $k = \text{ar}(\mathtt{w})$, *and* $\bar{x} = (x_1,\dots,x_k)$ *is a tuple of* k *pairwise distinct variables, then* $(s = \mathtt{w}(\bar{x}))$ *is a* formula.

(3) *If* φ *and* ψ *are formulas, then* $\neg\varphi$ *and* $(\varphi \vee \psi)$ *are also* formulas.

(4) *If* φ *is a formula and* $y \in \mathsf{vars}$, *then* $\exists y\, \varphi$ *is a* formula.

(5) *If* φ *is a formula,* $\mathtt{w} \in \mathbf{W}$, $S = \text{type}(\mathtt{w})$, $s \in S$, $k = \text{ar}(\mathtt{w})$, *and* $\bar{y} = (y_1,\dots,y_k)$ *is a tuple of* k *pairwise distinct variables, then* $\big(s = \sum \mathtt{w}(\bar{y}).\varphi\big)$ *is a* formula.

(6) *If* $\mathsf{P} \in \mathbb{P}$, $m = \text{ar}(\mathsf{P})$, *and* $t_1,\dots,t_m$ *are* $\mathbb{S}$*-terms such that* $(\text{type}(t_1),\dots,\text{type}(t_m)) = \text{type}(\mathsf{P})$, *then* $\mathsf{P}(t_1,\dots,t_m)$ *is a* formula.

(7) *For every* $S \in \mathbb{S}$ *and every* $s \in S$, s *is an* $\mathbb{S}$-term *of type* S.

(8) *For every* $S \in \mathbb{S}$, *every* $\mathtt{w} \in \mathbf{W}$ *of type* S, *and every tuple* $(x_1,\dots,x_k)$ *of* $k := \text{ar}(\mathtt{w})$ *pairwise distinct variables in* vars, $\mathtt{w}(x_1,\dots,x_k)$ *is an* $\mathbb{S}$-term *of type* S.

(9) *If* t_1 *and* t_2 *are* $\mathbb{S}$*-terms of the same type* S, *then so are* $(t_1{+}t_2)$ *and* $(t_1{-}t_2)$; *furthermore, if* S *is a ring (and not just an abelian group), then also* $(t_1{\cdot}t_2)$ *is an* $\mathbb{S}$*-term of type* S.

(10) *If* φ *is a formula,* $S \in \mathbb{S}$, *and* p *is a* $\mathbf{W}$*-product of type* S, *then* $\sum p.\varphi$ *is an* $\mathbb{S}$-term *of type* S.

Let $\mathcal{I} = (\mathcal{A},\beta)$ *be a* $(\sigma,\mathbf{W})$*-interpretation. For a formula or* $\mathbb{S}$*-term* ξ *of* $\text{FOWA}(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$, *the semantics* $\llbracket\xi\rrbracket^{\mathcal{I}}$ *is defined as follows.*

(1) $\llbracket x_1{=}x_2\rrbracket^{\mathcal{I}} = 1$ *if* $a_1{=}a_2$, *and* $\llbracket x_1{=}x_2\rrbracket^{\mathcal{I}} = 0$ *otherwise;*
$\llbracket R(x_1,\dots,x_{\text{ar}(R)})\rrbracket^{\mathcal{I}} = 1$ *if* $(a_1,\dots,a_{\text{ar}(R)}) \in R^{\mathcal{A}}$, *and* $\llbracket R(x_1,\dots,x_{\text{ar}(R)})\rrbracket^{\mathcal{I}} = 0$ *otherwise;*
where $a_j := \beta(x_j)$ *for* $j \in \{1,\dots,\max\{2,\text{ar}(R)\}\}$.

(2) $\llbracket(s = \mathtt{w}(\bar{x}))\rrbracket^{\mathcal{I}} = 1$ *if* $s = \mathtt{w}^{\mathcal{A}}(\beta(x_1),\dots,\beta(x_k))$, *and* $\llbracket(s = \mathtt{w}(\bar{x}))\rrbracket^{\mathcal{I}} = 0$ *otherwise.*

(3) $\llbracket\neg\varphi\rrbracket^{\mathcal{I}} = 1 - \llbracket\varphi\rrbracket^{\mathcal{I}}$ *and* $\llbracket(\varphi \vee \psi)\rrbracket^{\mathcal{I}} = \max\{\llbracket\varphi\rrbracket^{\mathcal{I}}, \llbracket\psi\rrbracket^{\mathcal{I}}\}$.

(4) $\llbracket\exists y\, \varphi\rrbracket^{\mathcal{I}} = \max\{\llbracket\varphi\rrbracket^{\mathcal{I}\frac{a}{y}} : a \in A\}$.

(5) $\llbracket\big(s = \sum \mathtt{w}(\bar{y}).\varphi\big)\rrbracket^{\mathcal{I}} = 1$ *if* $s = \sum_S\{\mathtt{w}^{\mathcal{A}}(\bar{a}) : \bar{a} = (a_1,\dots,a_k) \in A^k$ *with* $\llbracket\varphi\rrbracket^{\mathcal{I}\frac{a_1,\dots,a_k}{y_1,\dots,y_k}} = 1\}$
(as usual, by convention, we let $\sum_S X = 0_S$ *if* $X = \emptyset$*).*

(6) $\llbracket\mathsf{P}(t_1,\dots,t_m)\rrbracket^{\mathcal{I}} = 1$ *if* $\big(\llbracket t_1\rrbracket^{\mathcal{I}},\dots,\llbracket t_m\rrbracket^{\mathcal{I}}\big) \in \llbracket\mathsf{P}\rrbracket$, *and* $\llbracket\mathsf{P}(t_1,\dots,t_m)\rrbracket^{\mathcal{I}} = 0$ *otherwise.*

(7) $\llbracket s\rrbracket^{\mathcal{I}} = s$.

(8) $\llbracket\mathtt{w}(x_1,\dots,x_k)\rrbracket^{\mathcal{I}} = \mathtt{w}^{\mathcal{A}}(\beta(x_1),\dots,\beta(x_k))$.

(9) $\llbracket(t_1 * t_2)\rrbracket^{\mathcal{I}} = \llbracket t_1\rrbracket^{\mathcal{I}} *_S \llbracket t_2\rrbracket^{\mathcal{I}}$, *for* $* \in \{+,-,\cdot\}$.

[2] In particular, if $\text{ar}(R) = 0$, then $R()$ is a formula.

(10) $[\![\sum p.\varphi]\!]^{\mathcal{I}} = \sum_S \{ [\![p]\!]^{\mathcal{I}\frac{a_1,\dots,a_k}{y_1,\dots,y_k}} : a_1,\dots,a_k \in A$ *with* $[\![\varphi]\!]^{\mathcal{I}\frac{a_1,\dots,a_k}{y_1,\dots,y_k}} = 1\}$, *where* $\{y_1,\dots,y_k\} = \mathrm{vars}(p)$ *and* $k = |\mathrm{vars}(p)|$ *and* $[\![p]\!]^{\mathcal{I}} = [\![t_1]\!]^{\mathcal{I}} \cdot_S \cdots \cdot_S [\![t_\ell]\!]^{\mathcal{I}}$ *if* $p = t_1 \cdot \cdots \cdot t_\ell$ *is of type* S.

An *expression* is a formula or an $\mathbb{S}$-term. As usual, for a formula φ and a $(\sigma, \mathbf{W})$-interpretation $\mathcal{I}$, we will often write $\mathcal{I} \models \varphi$ to indicate that $[\![\varphi]\!]^{\mathcal{I}} = 1$. Accordingly, $\mathcal{I} \not\models \varphi$ indicates that $[\![\varphi]\!]^{\mathcal{I}} = 0$.

The set $\mathrm{vars}(\xi)$ of an expression ξ is defined as the set of all variables in vars that occur in ξ. The *free variables* $\mathrm{free}(\xi)$ of ξ are defined as follows: $\mathrm{free}(\xi) = \mathrm{vars}(\xi)$ if ξ is built according to one of the rules (1), (2), (7), (8); $\mathrm{free}(\neg\varphi) = \mathrm{free}(\varphi)$, $\mathrm{free}((\varphi \vee \psi)) = \mathrm{free}(\varphi) \cup \mathrm{free}(\psi)$, $\mathrm{free}(\exists y\, \varphi) = \mathrm{free}(\varphi) \setminus \{y\}$, $\mathrm{free}((s = \sum \mathtt{w}(y_1,\dots,y_k).\varphi)) = \mathrm{free}(\varphi) \setminus \{y_1,\dots,y_k\}$; $\mathrm{free}(\mathsf{P}(t_1,\dots,t_m)) = \bigcup_{i=1}^m \mathrm{free}(t_i)$; $\mathrm{free}((t_1 * t_2)) = \mathrm{free}(t_1) \cup \mathrm{free}(t_2)$ for $* \in \{+,-,\cdot\}$; $\mathrm{free}(\sum p.\varphi) = \mathrm{free}(\varphi) \setminus \mathrm{vars}(p)$. As usual, we will write $\xi(\bar{x})$ for $\bar{x} = (x_1,\dots,x_k)$ to indicate that $\mathrm{free}(\xi) \subseteq \{x_1,\dots,x_k\}$. A *sentence* is a $\mathrm{FOWA}(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$-formula φ with $\mathrm{free}(\varphi) = \emptyset$. A *ground* $\mathbb{S}$*-term* is an $\mathbb{S}$-term t of $\mathrm{FOWA}(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$ with $\mathrm{free}(t) = \emptyset$.

For a $(\sigma, \mathbf{W})$-structure $\mathcal{A}$ and a tuple $\bar{a} = (a_1,\dots,a_k) \in A^k$, we write $\mathcal{A} \models \varphi[\bar{a}]$ or $(\mathcal{A},\bar{a}) \models \varphi$ to indicate that for every assignment $\beta\colon \mathsf{vars} \to A$ with $\beta(x_i) = a_i$ for all $i \in [k]$, we have $\mathcal{I} \models \varphi$, for $\mathcal{I} = (\mathcal{A},\beta)$. Similarly, for an $\mathbb{S}$-term $t(\bar{x})$ we write $t^{\mathcal{A}}[\bar{a}]$ to denote $[\![t]\!]^{\mathcal{I}}$.

▶ **Definition 3.4** (FOWA_1 and FOW_1)**.** *The set of* formulas *and* $\mathbb{S}$-terms *of the logic* $\mathrm{FOWA}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$ *is built according to the same rules as for the logic* $\mathrm{FOWA}(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$, *with the following restrictions:*
$(5)_1$*: rule* (5) *can only be applied if* S *is* finite*,*
$(6)_1$*: rule* (6) *can only be applied if* $|\mathrm{free}(t_1) \cup \cdots \cup \mathrm{free}(t_m)| \leqslant 1$.
$\mathrm{FOW}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$ *is the restriction of* $\mathrm{FOWA}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$ *where rule* (10) *cannot be applied.*

Note that first-order logic $\mathrm{FO}[\sigma]$ is the restriction of $\mathrm{FOW}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$ where only rules (1), (3), and (4) can be applied. As usual, we write $(\varphi \wedge \psi)$ and $\forall y\, \varphi$ as shorthands for $\neg(\neg\varphi \vee \neg\psi)$ and $\neg\exists y\, \neg\varphi$. The *quantifier rank* $\mathrm{qr}(\xi)$ of a $\mathrm{FOWA}(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$-expression ξ is defined as the maximum nesting depth of constructs using rules (4) and (5) in order to construct ξ. The *aggregation depth* $\mathrm{d_{ag}}(\xi)$ of ξ is defined as the maximum nesting depth of term constructions using rule (10) in order to construct ξ.

▶ Remark 3.5. FOW_1 can be viewed as an extension of first-order logic with modulo-counting quantifiers: if $\mathbb{S}$ contains the abelian group $(\mathbb{Z}/m\mathbb{Z}, +)$ for some $m \geqslant 2$, and $\mathbf{W}$ contains a unary weight symbol $\mathtt{one}_m$ of type $\mathbb{Z}/m\mathbb{Z}$ such that $\mathtt{one}_m^{\mathcal{A}}(a) = 1$ for all $a \in A$, then the modulo m counting quantifier $\exists^{i \bmod m} y\, \varphi$ (stating that the number of interpretations for y that satisfy φ is congruent to i modulo m) can be expressed in $\mathrm{FOW}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$ via $\big(i = \sum \mathtt{one}_m(y).\varphi\big)$.

FOWA_1 can be viewed as an extension of the logic FOC_1 of [8]: if $\mathbb{S}$ contains the integer ring $(\mathbb{Z},+,\cdot)$ and $\mathbf{W}$ contains a unary weight symbol $\mathtt{one}$ of type $\mathbb{Z}$ such that $\mathtt{one}^{\mathcal{A}}(a) = 1$ for all $a \in A$ on all considered $(\sigma,\mathbf{W})$-structures $\mathcal{A}$, then the counting term $\#(y_1,\dots,y_k).\varphi$ of FOC_1 (which counts the number of tuples $(y_1,\dots,y_k)$ that satisfy φ) can be expressed in $\mathrm{FOWA}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$ via the $\mathbb{S}$-term $\sum p.\varphi$ for $p := \mathtt{one}(y_1) \cdot \cdots \cdot \mathtt{one}(y_k)$.

Let us mention, again, that we have designed the precise definition of the syntax of our logic in a way particularly suitable for formulating and proving the locality results that are crucial for obtaining our learning results. To obtain a more user-friendly syntax, i.e. which allows to read and construct formulas in a more intuitive way, it would of course make sense to introduce syntactic sugar that allows to explicitly write statements of the form

- $\#(y_1,\dots,y_k).\varphi$ instead of $\sum p.\varphi$ for $p := \mathtt{one}(y_1) \cdot \cdots \cdot \mathtt{one}(y_k)$
- $(\#(y).\varphi \equiv i \bmod m)$ or $\exists^{i \bmod m} y\, \varphi$ instead of $\big(i = \sum \mathtt{one}_m(y).\varphi\big)$.

For this, one would tacitly assume that $\mathbb{S}$ contains $(\mathbb{Z}, +, \cdot)$ (or $(\mathbb{Z}/m\mathbb{Z}, +)$) and $\mathbf{W}$ contains a unary weight symbol $\texttt{one}$ of type $\mathbb{Z}$ (or $\texttt{one}_m$ of type $\mathbb{Z}/m\mathbb{Z}$) where $\texttt{one}^{\mathcal{A}}(a) = 1$ ($= \texttt{one}^{\mathcal{A}}_m(a)$) for every $a \in A$ and every considered $(\sigma, \mathbf{W})$-structure $\mathcal{A}$.

To close this section, we return to the running examples from Examples 3.1 and 3.2.

▶ **Example 3.6.** We use the syntactic sugar introduced at the end of Remark 3.5.

(a) The number of consumers who bought products p from the product group defined by $\varphi_{\text{group}}(p)$ is specified by the $\mathbb{S}$-term

$$t_{\#\text{cons}} := \sum \texttt{one}(c)\ .\ \exists i\, \exists p\, \exists r\, (\varphi_{\text{group}}(p) \wedge T(i, c, p, r));$$

and using the syntactic sugar described above, this $\mathbb{S}$-term can be expressed via $\#(c).\ \exists i\, \exists p\, \exists r\, (\varphi_{\text{group}}(p) \wedge T(i, c, p, r))$.

The consumers c who spent at least as much as the *average consumer* on the products p satisfying $\varphi_{\text{group}}(p)$ can be described by the formula

$$\varphi_{\text{spending}}(c) := \ \mathsf{P}_{\geqslant}\big((t_{\text{spending}}(c) \cdot t_{\#\text{cons}})\ ,\ t_{\text{sales}}\big),$$

where $\mathsf{P}_{\geqslant}$ is a binary predicate in $\mathbb{P}$ of type $\mathbb{Q} \times \mathbb{Q}$ that is interpreted by the $\geqslant$-relation. To improve readability, one could introduce syntactic sugar that allows to express this as $t_{\text{spending}}(c) \geqslant t_{\text{sales}}/t_{\#\text{cons}}$. The formula $\varphi_{\text{spending}}(c)$ belongs to $\text{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$.

(b) The term $t_{\#\text{follows}}(x) := \#(y).F(x, y)$ specifies the number of users y followed by person x. The term $t_{\text{sum}}(x) := \sum \texttt{embedding(y)}.F(x, y)$ specifies the sum of the vectors associated with all users y followed by x. To describe the users x whose embedding is δ-close (for some fixed $\delta > 0$) to the average of the embeddings of users they follow[3], we might want to use a formula $\varphi_{\text{close}}(x)$ of the form

$$d\,\Big(\,\texttt{embedding}(x)\ ,\ \tfrac{1}{t_{\#\text{follows}}(x)} \cdot t_{\text{sum}}(x)\,\Big)\ <\ \delta\,.$$

We can describe this in $\text{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ by the formula

$$\varphi_{\text{close}}(x)\ :=\ \mathsf{P}_{\text{dist}<\delta}(\texttt{embedding}(x), t_{\#\text{follows}}(x), t_{\text{sum}}(x)),$$

where $\mathsf{P}_{\text{dist}<\delta}$ is a ternary predicate in $\mathbb{P}$ of type $\mathbb{R}^k \times \mathbb{Z} \times \mathbb{R}^k$ consisting of all triples $(\bar{v}, \ell, \bar{w})$ with $\ell > 0$ and $d(\bar{v}, \frac{1}{\ell}\cdot\bar{w}) < \delta$.

(c) Recall the term $t_B(x)$ introduced in Example 3.2 (c) that specifies the sum of the weights of edges between x and its blue neighbours, and let $t_R(x)$ be a similar term summing up the weights of edges between x and its red neighbours (using the syntactic sugar introduced at the end of Example 3.2, this can be described as $\sum_y \texttt{w}(x, y).(E(x, y) \wedge R(y)))$.

To specify the vertices x that have exactly 5 red neighbours, we can use the formula $\varphi_{5\,\text{red}}(x) := (\,5 = \#(y).(E(x, y) \wedge R(y))\,)$. Let us now assume we are given a particular set $H \subseteq \mathbb{R}^{2k}$ and we want to specify the vertices x that have exactly 5 red neighbours and for which, in addition, the $2k$-ary vector obtained by concatenating the k-ary vectors computed by summing up the weights of edges between x and its blue neighbours and by summing up the weights of edges between x and its red neighbours belongs to H. To express this, we can use a binary predicate P of type $\mathbb{R}^k \times \mathbb{R}^k$ with $[\![\mathsf{P}]\!] = \big\{(\bar{u}, \bar{v}) \in \mathbb{R}^k \times \mathbb{R}^k\ :\ (u_1, \ldots, u_k, v_1, \ldots, v_k) \in H\big\}$. Then, the $\text{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$-formula $\psi(x) := \varphi_{5\,\text{red}}(x) \wedge \mathsf{P}(t_B(x), t_G(x))$ specifies the vertices x we are interested in.

[3] Depending on the target of the embeddings, this could mean that the user mostly follows users with a very similar personality or political leaning.

4 Locality Properties of FOW_1 and FOWA_1

We now summarise locality properties of FOW_1 and FOWA_1 that are similar to well-known locality properties of first-order logic FO and to locality properties of FOC_1 achieved in [8]. This includes *Feferman-Vaught decompositions* and a *Gaifman normal form* for FOW_1, and a *localisation theorem* for the more expressive logic FOWA_1.

For the remainder of this section, let us fix a signature σ, a collection $\mathbb{S}$ of rings and/or abelian groups, a finite set $\mathbf{W}$ of weight symbols, and an $\mathbb{S}$-predicate collection $(\mathbb{P}, \mathrm{ar}, \mathrm{type}, [\![\cdot]\!])$.

The notion of *local formulas* is defined as usual [15]: let $r \in \mathbb{N}$. A $\mathrm{FOWA}(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$-formula $\varphi(\bar{x})$ with free variables $\bar{x} = (x_1, \dots, x_k)$ is *r-local (around $\bar{x}$)* if for every $(\sigma, \mathbf{W})$-structure $\mathcal{A}$ and all $\bar{a} \in A^k$, we have $\mathcal{A} \models \varphi[\bar{a}] \iff \mathcal{N}_r^{\mathcal{A}}(\bar{a}) \models \varphi[\bar{a}]$. A formula is *local* if it is r-local for some $r \in \mathbb{N}$.

For an $r \in \mathbb{N}$, it is straightforward to construct an $\mathrm{FO}[\sigma]$-formula $\mathrm{dist}^{\sigma}_{\leqslant r}(x, y)$ such that for every $(\sigma, \mathbf{W})$-structure $\mathcal{A}$ and all $a, b \in A$, we have $\mathcal{A} \models \mathrm{dist}^{\sigma}_{\leqslant r}[a, b] \iff \mathrm{dist}^{\mathcal{A}}(a, b) \leqslant r$. To improve readability, we write $\mathrm{dist}^{\sigma}(x, y) \leqslant r$ for $\mathrm{dist}^{\sigma}_{\leqslant r}(x, y)$, and $\mathrm{dist}^{\sigma}(x, y) > r$ for $\neg\,\mathrm{dist}^{\sigma}_{\leqslant r}(x, y)$; and we omit the superscript σ when it is clear from the context. For a tuple $\bar{x} = (x_1, \dots, x_k)$ of variables, $\mathrm{dist}(\bar{x}, y) > r$ is a shorthand for $\bigwedge_{i=1}^{k} \mathrm{dist}(x_i, y) > r$, and $\mathrm{dist}(\bar{x}, y) \leqslant r$ is a shorthand for $\bigvee_{i=1}^{k} \mathrm{dist}(x_i, y) \leqslant r$. For $\bar{y} = (y_1, \dots, y_\ell)$, we use $\mathrm{dist}(\bar{x}; \bar{y}) > r$ and $\mathrm{dist}(\bar{x}; \bar{y}) \leqslant r$ as shorthands for $\bigwedge_{j=1}^{\ell} \mathrm{dist}(\bar{x}, y_j) > r$ and $\bigvee_{j=1}^{\ell} \mathrm{dist}(\bar{x}, y_j) \leqslant r$, respectively.

The *r-localisation* $\varphi^{(r)}$ of a $\mathrm{FOWA}(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$-formula $\varphi(\bar{x})$ is the formula obtained from φ by replacing every subformula of the form $\exists y\, \varphi'$ with the formula $\exists y\, \big(\varphi' \wedge \mathrm{dist}(\bar{x}, y) \leqslant r\big)$, replacing every subformula of the form $\big(s = \sum \mathtt{w}(\bar{y}).\varphi'\big)$, for $\bar{y} = (y_1, \dots, y_k)$, with the formula $\big(s = \sum \mathtt{w}(\bar{y}).(\varphi' \wedge \bigwedge_{j=1}^{k} \mathrm{dist}(\bar{x}, y_j) \leqslant r)\big)$, and replacing every $\mathbb{S}$-term of the form $\sum p.\varphi'$ with the $\mathbb{S}$-term $\sum p.\big(\varphi' \wedge \bigwedge_{j=1}^{k} \mathrm{dist}(\bar{x}, y_j) \leqslant r\big)$, where $\{y_1, \dots, y_k\} = \mathrm{free}(\varphi')$. The resulting formula $\varphi^{(r)}(\bar{x})$ is r-local.

Feferman-Vaught Decompositions for FOW_1. We pick two new unary relation symbols X, Y that do not belong to σ, and we let $\sigma' := \sigma \cup \{X, Y\}$.

▶ **Definition 4.1.** *Let $\mathcal{A}, \mathcal{B}$ be $(\sigma, \mathbf{W})$-structures with $A \cap B = \emptyset$. The* disjoint sum *$\mathcal{A} \oplus \mathcal{B}$ is the $(\sigma', \mathbf{W})$-structure $\mathcal{C}$ with universe $C = A \cup B$, $X^{\mathcal{C}} = A$, $Y^{\mathcal{C}} = B$, $R^{\mathcal{C}} = R^{\mathcal{A}} \cup R^{\mathcal{B}}$ for all $R \in \sigma$, and such that for all $\mathtt{w} \in \mathbf{W}$ and $k := \mathrm{ar}(\mathtt{w})$ and all $\bar{c} = (c_1, \dots, c_k) \in C^k$, we have $\mathtt{w}^{\mathcal{C}}(\bar{c}) = \mathtt{w}^{\mathcal{A}}(\bar{c})$ if $\bar{c} \in A^k$, $\mathtt{w}^{\mathcal{C}}(\bar{c}) = \mathtt{w}^{\mathcal{B}}(\bar{c})$ if $\bar{c} \in B^k$, and $\mathtt{w}^{\mathcal{C}}(\bar{c}) = 0_S$ otherwise (for $S := \mathrm{type}(\mathtt{w})$). The* disjoint union *$\mathcal{A} \sqcup \mathcal{B}$ is the $(\sigma, \mathbf{W})$-structure obtained from $\mathcal{C} := \mathcal{A} \oplus \mathcal{B}$ by omitting the relations $X^{\mathcal{C}}, Y^{\mathcal{C}}$.*

▶ **Definition 4.2.** *Let L be a subset of $\mathrm{FOWA}(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$.
Let $k, \ell \in \mathbb{N}$ and let $\bar{x} = (x_1, \dots, x_k)$, $\bar{y} = (y_1, \dots, y_\ell)$ be tuples of $k{+}\ell$ pairwise distinct variables. Let φ be a $\mathrm{FOWA}(\mathbb{P})[\sigma', \mathbb{S}, \mathbf{W}]$-formula with $\mathrm{free}(\varphi) \subseteq \{x_1, \dots, x_k, y_1, \dots, y_\ell\}$. A* Feferman-Vaught decomposition *of φ in L w.r.t. $(\bar{x}; \bar{y})$ is a finite, non-empty set Δ of tuples of the form $\big(\alpha, \beta\big)$ where $\alpha, \beta \in \mathrm{L}$ and $\mathrm{free}(\alpha) \subseteq \{x_1, \dots, x_k\}$ and $\mathrm{free}(\beta) \subseteq \{y_1, \dots, y_\ell\}$, such that the following is true for all $(\sigma, \mathbf{W})$-structures $\mathcal{A}, \mathcal{B}$ with $A \cap B = \emptyset$ and all $\bar{a} \in A^k$, $\bar{b} \in B^\ell$: $\mathcal{A} \oplus \mathcal{B} \models \varphi[\bar{a}, \bar{b}] \iff$ there exists $(\alpha, \beta) \in \Delta$ such that $\mathcal{A} \models \alpha[\bar{a}]$ and $\mathcal{B} \models \beta[\bar{b}]$.*

Our first main result provides Feferman-Vaught decompositions for FOW_1.

▶ **Theorem 4.3** (Feferman-Vaught decompositions for $\mathrm{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$).
Let $k, \ell \in \mathbb{N}$ and let $\bar{x} = (x_1, \dots, x_k)$, $\bar{y} = (y_1, \dots, y_\ell)$ be tuples of $k{+}\ell$ pairwise distinct variables. For every $\mathrm{FOW}_1(\mathbb{P})[\sigma', \mathbb{S}, \mathbf{W}]$-formula φ with $\mathrm{free}(\varphi) \subseteq \{x_1, \dots, x_k, y_1, \dots, y_\ell\}$, there exists a Feferman-Vaught decomposition Δ in L of φ w.r.t. $(\bar{x}; \bar{y})$, where $\mathrm{L} := \mathrm{L}_\varphi$ is

the class of all $\mathrm{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$*-formulas of quantifier rank at most* $\mathrm{qr}(\varphi)$ *which use only those* $\mathsf{P} \in \mathbb{P}$ *and* $S \in \mathbb{S}$ *that occur in* φ *and only those* $\mathbb{S}$*-terms that occur in* φ *or that are of the form* s *for an* $s \in S \in \mathbb{S}$ *where* S *is finite and occurs in* φ*.*

Furthermore, there is an algorithm that computes Δ *upon input of* $\varphi, \bar{x}, \bar{y}$*.*

The proof proceeds in a similar way as the proof of the Feferman-Vaught decomposition for first-order logic with modulo-counting quantifiers in [14]. For details as well as for the proof of the following corollary of the theorem, we refer to [23].

▶ **Corollary 4.4.** *Let* $k, \ell \in \mathbb{N}$ *and let* $\bar{x} = (x_1, \ldots, x_k)$*,* $\bar{y} = (y_1, \ldots, y_\ell)$ *be tuples of* $k+\ell$ *pairwise distinct variables. Upon input of an* $r \in \mathbb{N}$ *and an* r*-local* $\mathrm{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$*-formula* $\varphi(\bar{x}, \bar{y})$*, one can compute a finite, non-empty set* Δ *of pairs* $\big(\alpha(\bar{x}), \beta(\bar{y})\big)$ *of* L*-formulas, where* L *is the class of all* r*-localisations of formulas in the class* L_φ *of Theorem 4.3, such that the following two formulas are equivalent:*

- $\Big(\bigwedge_{i=1}^{k} \bigwedge_{j=1}^{\ell} \mathrm{dist}(x_i, y_j) > 2r{+}1\Big) \ \wedge \ \varphi(\bar{x}, \bar{y})$
- $\Big(\bigwedge_{i=1}^{k} \bigwedge_{j=1}^{\ell} \mathrm{dist}(x_i, y_j) > 2r{+}1\Big) \ \wedge \ \bigvee_{(\alpha,\beta)\in\Delta} \big(\alpha(\bar{x}) \wedge \beta(\bar{y})\big)$*.*

Gaifman Normal Form for $\mathbf{FOW_1}$. We now turn to a Gaifman normal form for FOW_1.

▶ **Definition 4.5.** *A* basic-local sentence *in* $\mathrm{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ *is a sentence of the form* $\exists x_1 \cdots \exists x_\ell \, \big(\bigwedge_{1\leqslant i<j\leqslant \ell} \mathrm{dist}(x_i, x_j) > 2r \ \wedge \ \bigwedge_{i=1}^{\ell} \lambda(x_i)\big)$*, where* $\ell \in \mathbb{N}_{\geqslant 1}$*,* $r \in \mathbb{N}$*,* $\lambda(x)$ *is an* r*-local* $\mathrm{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$*-formula, and* $x_1, \ldots, x_\ell$ *are* ℓ *pairwise distinct variables.*

A local aggregation sentence *in* $\mathrm{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ *is a sentence of the form* $\big(s = \sum \mathtt{w}(\bar{y}).\lambda(\bar{y})\big)$*, where* $\mathtt{w} \in \mathbf{W}$*,* $s \in S := \mathrm{type}(\mathtt{w})$*,* $\ell = \mathrm{ar}(\mathtt{w})$*,* $\bar{y} = (y_1, \ldots, y_\ell)$ *is a tuple of* ℓ *pairwise distinct variables, and* $\lambda(\bar{y})$ *is an* r*-local* $\mathrm{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$*-formula.*

A $\mathrm{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$*-formula* in Gaifman normal form *is a Boolean combination of local* $\mathrm{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$*-formulas, basic-local sentences in* $\mathrm{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$*, and local aggregation sentences in* $\mathrm{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$*.*

Our next main theorem provides a Gaifman normal form for FOW_1.

▶ **Theorem 4.6** (Gaifman normal form for $\mathrm{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$)**.** *Every* $\mathrm{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$*-formula* φ *is equivalent to an* $\mathrm{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$*-formula* γ *in Gaifman normal form with* $\mathrm{free}(\gamma) = \mathrm{free}(\varphi)$*. Furthermore, there is an algorithm that computes* γ *upon input of* φ*.*

The proof proceeds similarly as Gaifman's original proof for first-order logic FO ([2], see also [4, Sect. 4.1]), but since subformulas are from $\mathrm{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$, we use Corollary 4.4 instead of Feferman-Vaught decompositions for FO (cf. [4, Lemma 2.3]). Furthermore, for formulas built according to rule $(5)_1$, we proceed in a similar way as for the modulo-counting quantifiers in the Gaifman normal construction of [14]. We defer the reader to the full version for the details [23].

A Localisation Theorem for $\mathbf{FOWA_1}$. Our next main theorem provides a locality result for the logic FOWA_1, which is a logic substantially more expressive than FOW_1.

▶ **Theorem 4.7** (Localisation Theorem for FOWA_1)**.** *For every* $\mathrm{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$*-formula* $\varphi(x_1, \ldots, x_k)$ *(with* $k \geqslant 0$*), there is an extension* σ_φ *of* σ *with relation symbols of arity* $\leqslant 1$*, and a* $\mathrm{FOW}_1(\mathbb{P})[\sigma_\varphi, \mathbb{S}, \mathbf{W}]$*-formula* $\varphi'(x_1, \ldots, x_k)$ *that is a Boolean combination of local formulas and statements of the form* $R()$ *where* $R \in \sigma_\varphi$ *has arity* 0*, for which the following* *is true: there is an algorithm*[4] *that, upon input of a* $(\sigma, \mathbf{W})$*-structure* $\mathcal{A}$*, computes in time* $|A|{\cdot}d^{\mathcal{O}(1)}$*, where* d *is the degree of* $\mathcal{A}$*, a* σ_φ*-expansion* $\mathcal{A}^\varphi$ *of* $\mathcal{A}$ *such that for all* $\bar{a} \in A^k$ *it holds that* $\mathcal{A}^\varphi \models \varphi'[\bar{a}] \iff \mathcal{A} \models \varphi[\bar{a}]$*.*

We prove this by decomposing FOWA_1-expressions into simpler expressions that can be evaluated in a structure $\mathcal{A}$ by exploring for each element a in the universe of $\mathcal{A}$ only a local neighbourhood around a. This is achieved by proving a decomposition theorem that is a generalisation of the decomposition for $\mathrm{FOC}_1(\mathbb{P})$ provided in [8, Theorem 6.6], and it builds upon the Gaifman normal form result of Theorem 4.6. Again, we defer the reader to the full version for the details [23].

5 Learning Concepts on Weighted Structures

Throughout this section, fix a collection $\mathbb{S}$ of rings and/or abelian groups, an $\mathbb{S}$-predicate collection $(\mathbb{P}, \mathrm{ar}, \mathrm{type}, [\![\cdot]\!])$, and a finite set $\mathbf{W}$ of weight symbols.

Furthermore, fix numbers $k, \ell \in \mathbb{N}$. Let L be a logic (e.g. FO, $\mathrm{FOW}_1(\mathbb{P})$, $\mathrm{FOWA}_1(\mathbb{P})$, $\mathrm{FOWA}(\mathbb{P})$), let σ be a signature, and let $\Phi \subseteq \mathrm{L}[\sigma, \mathbb{S}, \mathbf{W}]$ be a set of formulas $\varphi(\bar{x}, \bar{y})$ with $|\bar{x}| = k$ and $|\bar{y}| = \ell$. For a $(\sigma, \mathbf{W})$-structure $\mathcal{A}$, we follow the same approach as [3, 6, 7, 9, 22] and consider the instance space $\mathcal{X} = A^k$ and concepts from the concept class

$$\mathcal{C}(\Phi, \mathcal{A}, k, \ell) \;\coloneqq\; \big\{\; [\![\varphi(\bar{x}, \bar{y})]\!]^{\mathcal{A}}(\bar{x}, \bar{v}) \;:\; \varphi \in \Phi,\; \bar{v} \in A^\ell \;\big\},$$

where $[\![\varphi(\bar{x}, \bar{y})]\!]^{\mathcal{A}}(\bar{x}, \bar{v})$ is defined as the mapping from A^k to $\{0,1\}$ that maps $\bar{a} \in A^k$ to $[\![\varphi(\bar{a}, \bar{v})]\!]^{\mathcal{A}}$, which is 1 iff $\mathcal{A} \models \varphi[\bar{a}, \bar{v}]$. Given a training sequence $T = \big((\bar{a}_1, b_1), \ldots, (\bar{a}_t, b_t)\big)$ from $(A^k \times \{0,1\})^t$, we want to compute a hypothesis that consists of a formula φ and a tuple of parameters $\bar{v}$ and is, depending on the approach, consistent with the training sequence or probably approximately correct.

Instead of allowing random access to the background structure, we limit our algorithms to have only *local access.* That is, an algorithm may only interact with the structure via queries of the form "Is $\bar{a} \in R^{\mathcal{A}}$?", "Return $\mathtt{w}^{\mathcal{A}}(\bar{a})$" and "Return a list of all neighbours of a in the Gaifman graph of $\mathcal{A}$". Hence, in this model, algorithms are required to access new vertices only via neighbourhood queries of vertices they have already seen. This enables us to learn a concept from examples even if the background structure is too large to fit into the main memory. To obtain a reasonable running time, we intend to find algorithms that compute a hypothesis in sublinear time, measured in the size of the background structure. This local access model has already been studied for relational structures in [7, 22] for concepts definable in FO or in $\mathrm{FOCN}(\mathbb{P})$. Modifications of the local access model for strings and trees have been studied in [3, 6].

In many applications, the same background structure is used multiple times to learn different concepts. Hence, similar to the approaches in [3, 6], we allow a precomputation step to enrich the background structure with additional information. That is, instead of learning on a $(\sigma, \mathbf{W})$-structure $\mathcal{A}$, we use an enriched $(\sigma^*, \mathbf{W})$-structure $\mathcal{A}^*$, which has the same universe as $\mathcal{A}$, but $\sigma^* \supseteq \sigma$ contains additional relation symbols. The hypotheses we compute may make use of this additional information and thus, instead of representing them via formulas from the fixed set Φ, we consider a set Φ^* of formulas of signature σ^*. These formulas may even belong to a logic L^* different from L. We study the following learning problem.

[4] with $\mathbb{P}$- and $\mathbb{S}$-oracles, i.e., the operations $+_S, \cdot_S$ for $S \in \mathbb{S}$ and checking if a tuple belongs to $[\![\mathsf{P}]\!]$ for $\mathsf{P} \in \mathbb{P}$ can be done in constant time by referring to an oracle that provides us with the answers

▶ **Problem 5.1** (Exact Learning with Precomputation). Let $\Phi \subseteq \mathrm{L}[\sigma, \mathbb{S}, \mathbf{W}]$ and $\Phi^* \subseteq \mathrm{L}^*[\sigma^*, \mathbb{S}, \mathbf{W}]$ such that, for every $(\sigma, \mathbf{W})$-structure $\mathcal{A}$, there is a $(\sigma^*, \mathbf{W})$-structure $\mathcal{A}^*$ with $U(\mathcal{A}^*) = U(\mathcal{A})$ that satisfies $\mathcal{C}(\Phi, \mathcal{A}, k, \ell) \subseteq \mathcal{C}(\Phi^*, \mathcal{A}^*, k, \ell)$, i.e. every concept that can be defined on $\mathcal{A}$ using Φ can also be defined on $\mathcal{A}^*$ using Φ^*. The task is as follows.

Given a training sequence $T = \big((\bar{a}_1, b_1), \dots, (\bar{a}_t, b_t)\big) \in (A^k \times \{0,1\})^t$ and, for a $(\sigma, \mathbf{W})$-structure $\mathcal{A}$, local access to the associated $(\sigma^*, \mathbf{W})$-structure $\mathcal{A}^*$,

return a formula $\varphi^* \in \Phi^*$ and a tuple $\bar{v} \in A^\ell$ of parameters such that the hypothesis $[\![\varphi^*(\bar{x}, \bar{y})]\!]^{\mathcal{A}^*}(\bar{x}, \bar{v})$ is consistent with T, i.e. it maps $\bar{a}_i$ to b_i for every $i \in [t]$.

The algorithm may reject if there is no consistent classifier using a formula from Φ on $\mathcal{A}$.

Next, we examine requirements for Φ and Φ^* that help us solve Problem 5.1 efficiently. Following the approach presented in [7], to obtain algorithms that run in sublinear time, we study concepts that can be represented via a set of *local* formulas Φ with a finite set Φ^* of normal forms. Using Feferman-Vaught decompositions and the locality of the formulas, we can then limit the search space for the parameters to those that are in a certain neighbourhood of the training sequence. Recall that Φ is a set of formulas $\varphi(\bar{x}, \bar{y})$ in $\mathrm{L}[\sigma, \mathbb{S}, \mathbf{W}]$ with $|\bar{x}| = k$ and $|\bar{y}| = \ell$. In the following, we require Φ to have the following property.

▶ **Property 5.2.** There are a signature σ^*, a logic L^*, an $r \in \mathbb{N}$, and a finite set of r-local formulas $\Phi^* \subseteq \mathrm{L}^*[\sigma^*, \mathbb{S}, \mathbf{W}]$ such that the following hold.

(1) For every $(\sigma, \mathbf{W})$-structure $\mathcal{A}$, there is a $(\sigma^*, \mathbf{W})$-structure $\mathcal{A}^*$ with $U(\mathcal{A}^*) = U(\mathcal{A})$ such that, for every $\varphi(\bar{x}, \bar{y}) \in \Phi$, there is a $\varphi^*(\bar{x}, \bar{y}) \in \Phi^*$ with $\mathcal{A} \models \varphi[\bar{a}, \bar{b}] \iff \mathcal{A}^* \models \varphi^*[\bar{a}, \bar{b}]$ for all $\bar{a} \in A^k$, $\bar{b} \in A^\ell$.

(2) Every $\varphi^* \in \Phi^*$ has, for every partition $(\bar{z}; \bar{z}')$ of the free variables of φ^*, a Feferman-Vaught decomposition in Φ^* w.r.t. $(\bar{z}; \bar{z}')$.

(3) For all $\varphi_1^*, \varphi_2^* \in \Phi^*$, the set Φ^* contains formulas equivalent to $\neg\varphi_1^*$ and to $(\varphi_1^* \vee \varphi_2^*)$.

This property suffices to solve Problem 5.1:

▶ **Theorem 5.3** (Exact Learning with Precomputation). *There is an algorithm that solves Problem 5.1 with local access to a structure $\mathcal{A}^*$ associated with a structure $\mathcal{A}$ in time $f_{\Phi^*}(\mathcal{A}^*) \cdot \big(\log n + d + t\big)^{\mathcal{O}(1)}$, where $\mathcal{A}$, $\mathcal{A}^*$, Φ, and Φ^* are as described in Property 5.2, t is the number of training examples, n and d are the size and the degree of $\mathcal{A}^*$, and $f_{\Phi^*}(\mathcal{A}^*)$ is an upper bound on the time complexity of model checking for formulas in Φ^* on $\mathcal{A}^*$.*

We prove the theorem in Section 5.1.

Apart from exact learning with precomputation, we also study hypotheses that generalise well in the following sense. The *generalisation error* of a hypothesis $h\colon A^k \to \{0,1\}$ for a probability distribution $\mathcal{D}$ on $A^k \times \{0,1\}$ is

$$\mathrm{err}_{\mathcal{D}}(h) \;:=\; \Pr_{(\bar{a},b)\sim\mathcal{D}}(h(\bar{a}) \neq b).$$

We write $\mathrm{rat}(0,1)$ for the set of all rationals q with $0 < q < 1$. A hypothesis class $\mathcal{H} \subseteq \{0,1\}^{A^k}$ is *agnostically PAC-learnable* if there is a function $t_{\mathcal{H}}\colon \mathrm{rat}(0,1)^2 \to \mathbb{N}$ and a learning algorithm $\mathfrak{L}$ such that for all $\varepsilon, \delta \in \mathrm{rat}(0,1)$ and for every distribution $\mathcal{D}$ over $A^k \times \{0,1\}$, when running $\mathfrak{L}$ on a sequence T of $t_{\mathcal{H}}(\varepsilon, \delta)$ examples drawn i.i.d. from $\mathcal{D}$, it holds that

$$\Pr\left(\mathrm{err}_{\mathcal{D}}(\mathfrak{L}(T)) \leqslant \inf_{h\in\mathcal{H}} \mathrm{err}_{\mathcal{D}}(h) + \varepsilon\right) \geqslant 1 - \delta.$$

The following theorem, which we prove in Section 5.2, provides an agnostic PAC learning algorithm.

▶ **Theorem 5.4** (Agnostic PAC Learning with Precomputation). *Let $\mathcal{A}$, $\mathcal{A}^*$, and Φ^* be as in Property 5.2. There is an $s \in \mathbb{N}$ such that, given local access to $\mathcal{A}^*$, the hypothesis class $\mathcal{H} := \mathcal{C}(\Phi^*, \mathcal{A}^*, k, \ell)$ is agnostically PAC-learnable with $t_{\mathcal{H}}(\varepsilon, \delta) = s \cdot \left\lceil \frac{\log(n/\delta)}{\varepsilon^2} \right\rceil$ via an algorithm that, given $t_{\mathcal{H}}(\varepsilon, \delta)$ examples, returns a hypothesis of the form $(\varphi^*, \bar{v}^*)$ with $\varphi^* \in \Phi^*$ and $\bar{v}^* \in A^\ell$ in time $f_{\Phi^*}(\mathcal{A}^*) \cdot \big(\log n + d + \frac{1}{\varepsilon} + \log \frac{1}{\delta}\big)^{\mathcal{O}(1)}$ with only local access to $\mathcal{A}^*$, where n and d are the size and the degree of $\mathcal{A}^*$, and $f_{\Phi^*}(\mathcal{A}^*)$ is an upper bound on the time complexity of model checking for formulas in Φ^* on $\mathcal{A}^*$.*

The next remark establishes the crucial link between the learning results of this section and the locality results of Section 4: it shows that suitably chosen sets $\Phi \subseteq \mathrm{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ indeed have Property 5.2.

▶ Remark 5.5. Fix a $q \in \mathbb{N}$ and let $\Phi := \Phi_{q,k+\ell}$ be the set of all $\mathrm{FO}[\sigma]$-formulas φ of quantifier rank at most q and with free variables among $\{x_1, \dots, x_k, y_1, \dots, y_\ell\}$. By the well-known properties of first-order logic, Φ has Property 5.2 (e.g. via $L' := L = \mathrm{FO}$, $\sigma^* := \sigma$, and $\mathcal{A}^* := \mathcal{A}$; this is exactly the setting considered in [7]). By using the locality properties of FOW_1 and FOWA_1 from Section 4, we can apply a similar reasoning to $\mathrm{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ as to $\mathrm{FO}[\sigma]$: let the collections $\mathbb{P}$ and $\mathbb{S}$ be finite (but $\mathbb{S}$ may contain some infinite rings or abelian groups), fix a finite set $\mathcal{S}$ of elements $s \in S \in \mathbb{S}$, and fix a $q \in \mathbb{N}$. Let $\Phi := \Phi_{q,k+\ell,\mathcal{S}}$ be the set of all $\mathrm{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$-formulas φ of quantifier rank and aggregation depth at most q and with free variables among $\{x_1, \dots, x_k, y_1, \dots, y_\ell\}$ that have the following additional property: all symbols $s \in S \in \mathbb{S}$ that are present in φ belong to $\mathcal{S}$, all $\mathbf{W}$-products present in φ have length at most q, and the maximum nesting depth of term constructions using rule (9) in order to construct terms present in φ is at most q. This set Φ has Property 5.2. To see why, note that up to logical equivalence, Φ only contains a finite number of formulas. For each of these finitely many formulas φ, we apply Theorem 4.7 to obtain an extension σ_φ of σ, a σ_φ-expansion $\mathcal{A}^\varphi$ of $\mathcal{A}$, and a local $\mathrm{FOW}_1(\mathbb{P})[\sigma_\varphi, \mathbb{S}, \mathbf{W}]$-formula φ'. Then we let σ^* be the union of all the σ_φ, we let $\mathcal{A}^*$ be the σ^*-expansion of $\mathcal{A}$ whose σ_φ-reduct coincides with $\mathcal{A}^\varphi$ for each φ, and we let Φ' be the set of all the formulas φ'. Choose a number $r \in \mathbb{N}$ such that each of the $\varphi' \in \Phi'$ is r-local. Now we can repeatedly apply Theorem 4.3, take the r-localisations $\alpha^{(r)}, \beta^{(r)}$ of the resulting formulas α, β, and take Boolean combinations to obtain a finite extension Φ^* of Φ' such that Φ^* satisfies statements (2) and (3) of Property 5.2 and contains only r-local formulas (see [23] for details on this construction). This Φ^* witnesses that $\Phi := \Phi_{q,k+\ell,\mathcal{S}}$ has Property 5.2.

5.1 Exact Learning with Precomputation

Section 5.1 is devoted to the proof of Theorem 5.3.

Let $\mathcal{A}$ be a $(\sigma, \mathbf{W})$-structure and let $\mathcal{A}^*$ and Φ^* be as in Property 5.2. To prove Theorem 5.3, we present an algorithm that follows similar ideas as the algorithm presented in [7]. Note, however, that [7] focuses on first-order logic, whereas our setting allows to achieve results for considerably stronger logics.

While the set of possible formulas Φ^* already has constant size, we have to reduce the parameter space to obtain an algorithm that runs in sublinear time. Since the formulas in Φ^* are r-local, we show that it suffices to consider parameters in a neighbourhood of the training sequence with a fixed radius.

For $S \subseteq A$ and an element $b \in A$, let $\mathrm{dist}^{\mathcal{A}^*}(b, S) := \min_{a \in S} \mathrm{dist}^{\mathcal{A}^*}(b, a)$. For $R \geqslant 0$, set $N_R^{\mathcal{A}^*}(S) := \bigcup_{a \in S} N_R^{\mathcal{A}^*}(a)$. Also, for a training sequence $T = \big((\bar{a}_1, b_1), \dots, (\bar{a}_t, b_t)\big) \in (A^k \times \{0,1\})^t$, let $N_R^{\mathcal{A}^*}(T) := N_R^{\mathcal{A}^*}(S)$, where S is the set of all $a \in A$ that occur in one of the $\bar{a}_i$.

```
 1: $N \leftarrow N^{\mathcal{A}^*}_{(2r+1)\ell}(T)$
 2: for all $\bar{v}^* \in N^\ell$ do
 3:     for all $\varphi^*(\bar{x},\bar{y}) \in \Phi^*$ do
 4:         $consistent \leftarrow$ true
 5:         for all $i \in [t]$ do
 6:             $\mathcal{N} = \mathcal{N}^{\mathcal{A}^*}_r(\bar{a}_i\bar{v}^*)$
 7:             if $[\![\varphi^*(\bar{a}_i,\bar{v}^*)]\!]^{\mathcal{N}} \neq b_i$ then
 8:                 $consistent \leftarrow$ false
 9:         if $consistent$ then
10:             return $(\varphi^*,\ \bar{v}^*)$
11: reject
```

```
 1: $N \leftarrow N^{\mathcal{A}^*}_{(2r+1)\ell}(T)$
 2: $err_{\text{min}} \leftarrow |T| + 1$
 3: for all $\bar{v}^* \in N^\ell$ do
 4:     for all $\varphi^*(\bar{x},\bar{y}) \in \Phi^*$ do
 5:         $err_{\text{cur}} \leftarrow 0$
 6:         for all $i \in [t]$ do
 7:             $\mathcal{N} = \mathcal{N}^{\mathcal{A}^*}_r(\bar{a}_i\bar{v}^*)$
 8:             if $[\![\varphi^*(\bar{a}_i,\bar{v}^*)]\!]^{\mathcal{N}} \neq b_i$ then
 9:                 $err_{\text{cur}} \leftarrow err_{\text{cur}} + 1$
10:         if $err_{\text{cur}} < err_{\text{min}}$ then
11:             $err_{\text{min}} \leftarrow err_{\text{cur}}$
12:             $\varphi^*_{\text{min}} \leftarrow \varphi^*$
13:             $\bar{v}^*_{\text{min}} \leftarrow \bar{v}^*$
14: return $(\varphi^*_{\text{min}},\ \bar{v}^*_{\text{min}})$
```

Figure 1 Learning algorithms for Theorems 5.3 (left) and 5.4 (right). Both algorithms use as input a training sequence $T = \big((\bar{a}_1, b_1), \dots, (\bar{a}_t, b_t)\big) \in (A^k \times \{0,1\})^t$ and have only local access to the structure $\mathcal{A}^*$.

▶ **Lemma 5.6.** *Let $T = \big((\bar{a}_1, b_1), \dots, (\bar{a}_t, b_t)\big) \in (A^k \times \{0,1\})^t$ be consistent with some classifier in $\mathcal{C}(\Phi^*, \mathcal{A}^*, k, \ell)$. Then there are a formula $\varphi^*(\bar{x},\bar{y}) \in \Phi^*$ and a tuple $\bar{v}^* \in N^{\mathcal{A}^*}_{(2r+1)\ell}(T)^\ell$ such that $[\![\varphi^*(\bar{x},\bar{y})]\!]^{\mathcal{A}^*}(\bar{x},\bar{v}^*)$ is consistent with T.*

The proof can be found in [23]. It is similar to the proof of the analogous statement in [7] for the special case of FO, but relies on Property 5.2. We can now prove Theorem 5.3.

Proof of Theorem 5.3. We show that the algorithm depicted on the left-hand side of Figure 1 fulfils the requirements given in Theorem 5.3. The algorithm goes through all tuples $\bar{v}^* \in (N^{\mathcal{A}^*}_{(2r+1)\ell}(T))^\ell$ and all formulas $\varphi^*(\bar{x},\bar{y}) \in \Phi^*$. A hypothesis $[\![\varphi^*(\bar{x},\bar{y})]\!]^{\mathcal{A}^*}(\bar{x},\bar{v}^*)$ is consistent with the training sequence T if and only if $[\![\varphi^*(\bar{a}_i,\bar{v}^*)]\!]^{\mathcal{A}^*} = b_i$ for all $i \in [t]$. Since Φ^* only contains r-local formulas, this holds if and only if $[\![\varphi^*(\bar{a}_i,\bar{v}^*)]\!]^{\mathcal{N}^{\mathcal{A}^*}_r(\bar{a}_i\bar{v}^*)} = b_i$ for every $i \in [t]$. Hence, the algorithm only returns a hypothesis if it is consistent. Furthermore, if there is a consistent hypothesis in $\mathcal{C}(\Phi, \mathcal{A}, k, \ell)$, then by Property 5.2 (1), there is also a consistent hypothesis in $\mathcal{C}(\Phi^*, \mathcal{A}^*, k, \ell)$, and Lemma 5.6 ensures that the algorithm then returns a hypothesis.

It remains to show that the algorithm satisfies the running time requirements while only using local access to the structure $\mathcal{A}^*$. For all $\bar{a} \in A^k$ and $\bar{v}^* \in A^\ell$, we can bound the size of their neighbourhood by $\big|N^{\mathcal{A}^*}_r(\bar{a}\bar{v}^*)\big| \leqslant (k+\ell) \cdot \sum_{i=0}^{r} d^i \leqslant (k+\ell) \cdot (1 + d^{r+1})$. Therefore, the representation size of the substructure $\mathcal{N}^{\mathcal{A}^*}_r(\bar{a}\bar{v}^*)$ is in $\mathcal{O}\big((k+\ell) \cdot d^{r+1} \cdot \log n\big)$. Thus, the consistency check in lines 4–8 runs in time $f_{\Phi^*}(\mathcal{A}^*) \cdot t \cdot \mathcal{O}\big((k+\ell) \cdot d^{r+1} \cdot \log n\big)$. The algorithm checks up to $|N|^\ell \cdot |\Phi^*| \in \mathcal{O}\big((tkd^{(2r+1)\ell+1})^\ell \cdot |\Phi^*|\big)$ hypotheses with $N = N^{\mathcal{A}^*}_{(2r+1)\ell}(T)$. All in all, since k, ℓ, r are considered constant, the running time of the algorithm is in $f_{\Phi^*}(\mathcal{A}^*) \cdot (\log n + d + t)^{\mathcal{O}(1)}$ and it only uses local access to the structure $\mathcal{A}^*$. ◀

5.2 Agnostic PAC Learning with Precomputation

Section 5.2 is devoted to the proof of Theorem 5.4.

To obtain a hypothesis that generalises well, we follow the *Empirical Risk Minimization* rule (ERM) [19, 24], i.e. our algorithm should return a hypothesis h that minimises the *training error*

$$\mathrm{err}_T(h) \ := \ \tfrac{1}{|T|} \cdot |\{(\bar{a}, b) \in T : h(\bar{a}) \neq b\}|$$

on the training sequence T. To prove Theorem 5.4, we use the following result from [19].

▶ **Lemma 5.7** (Uniform Convergence [19]). *Let $\mathcal{H}$ be a finite class of hypotheses $h\colon A^k \to \{0,1\}$. Then $\mathcal{H}$ is agnostically PAC-learnable using an ERM algorithm and*

$$t_{\mathcal{H}}(\varepsilon, \delta) \ := \ \left\lceil \frac{2\log(2\,|\mathcal{H}|\,/\delta)}{\varepsilon^2} \right\rceil .$$

Proof of Theorem 5.4. We show that the algorithm depicted on the right-hand side of Figure 1 fulfils the requirements from Theorem 5.4. The algorithm goes through all tuples $\bar{v}^* \in (N^{\mathcal{A}^*}_{(2r+1)\ell}(T))^\ell$ and all formulas $\varphi^*(\bar{x}, \bar{y}) \in \Phi^*$ and counts the number of errors that $[\![\varphi^*(\bar{x}, \bar{y})]\!]^{\mathcal{A}^*}(\bar{x}, \bar{v}^*)$ makes on T. Then it returns the hypothesis with the minimal training error.

Since Φ^* and A^ℓ are finite, $\mathcal{H} = \mathcal{C}(\Phi^*, \mathcal{A}^*, k, \ell)$ is finite. Thus, using Lemma 5.7, $\mathcal{H}$ is agnostically PAC-learnable with $t_{\mathcal{H}}(\varepsilon, \delta) = \left\lceil \frac{2\log(2|\mathcal{H}|/\delta)}{\varepsilon^2} \right\rceil \leqslant \left\lceil \frac{4\ell\log(|\Phi^*|)\log(n/\delta)}{\varepsilon^2} \right\rceil$. The running time analysis works as in the proof of Theorem 5.3. The algorithm returns a hypothesis in time $f_{\Phi^*}(\mathcal{A}^*) \cdot (\log n + d + t)^{\mathcal{O}(1)}$. For a training sequence of length $t = t_{\mathcal{H}}(\varepsilon, \delta)$, we obtain a running time in $f_{\Phi^*}(\mathcal{A}^*) \cdot \big(\log n + d + \log(1/\delta) + 1/\varepsilon\big)^{\mathcal{O}(1)}$. ◀

6 Putting Things Together

Let the collections $\mathbb{P}$ and $\mathbb{S}$ be finite (but $\mathbb{S}$ may contain infinite rings or abelian groups), fix a *finite* set $\mathcal{S}$ of elements $s \in S \in \mathbb{S}$, fix a $q \in \mathbb{N}$, and let $\Phi := \Phi_{q,k+\ell,\mathcal{S}}$ be the set of $\mathrm{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$-formulas defined in Remark 5.5. Let Φ^*, σ^*, and $\mathcal{A}^*$ (for all $(\sigma, \mathbf{W})$-structures $\mathcal{A}$) be as described in Remark 5.5. By Theorem 4.7, $\mathcal{A}^*$ can be computed from $\mathcal{A}$ in time $|A|{\cdot}d^{\mathcal{O}(1)}$, where d is the degree of $\mathcal{A}$. By Remark 5.5, the formulas in Φ^* are r-local for a fixed number r, and this implies that model checking for a formula in Φ^* on $\mathcal{A}^*$ can be done in time polynomial in d. Combining this with Theorems 5.3 and 5.4 yields the following[5].

▶ **Theorem 6.1.** *Let n and d denote the size and the degree of $\mathcal{A}$.*

(1) *There is an algorithm that solves Exact Learning with Precomputation for Φ and Φ^* with local access to a structure $\mathcal{A}^*$ associated with a structure $\mathcal{A}$ in time $(\log n + d + t)^{\mathcal{O}(1)}$, where t is the number of training examples.*

(2) *There is an $s \in \mathbb{N}$ such that, given local access to a structure $\mathcal{A}^*$ associated with a structure $\mathcal{A}$, the hypothesis class $\mathcal{H} := \mathcal{C}(\Phi^*, \mathcal{A}^*, k, \ell)$ is agnostically PAC-learnable with $t_{\mathcal{H}}(\varepsilon, \delta) = s \cdot \left\lceil \frac{\log(n/\delta)}{\varepsilon^2} \right\rceil$ via an algorithm that, given $t_{\mathcal{H}}(\varepsilon, \delta)$ examples, returns a hypothesis of the form $(\varphi^*, \bar{v}^*)$ with $\varphi^* \in \Phi^*$ and $\bar{v}^* \in A^\ell$ in time $\big(\log n + d + \frac{1}{\varepsilon} + \log\frac{1}{\delta}\big)^{\mathcal{O}(1)}$ with only local access to $\mathcal{A}^*$.*

Additionally, the algorithms can be chosen such that the returned hypotheses can be evaluated in time $(\log n + d)^{\mathcal{O}(1)}$.

[5] All mentioned algorithms are assumed to have $\mathbb{P}$- and $\mathbb{S}$-oracles, so that operations $+_S, \cdot_S$ for $S \in \mathbb{S}$ and checking if a tuple is in $[\![\mathsf{P}]\!]$ for $\mathsf{P} \in \mathbb{P}$ takes time $\mathcal{O}(1)$.

We conclude with an example that illustrates an application scenario for Theorem 6.1.

▶ **Example 6.2.** Recall the $(\sigma, \mathbf{W})$-structure $\mathcal{A}$ for the online marketplace from part (a) of Examples 3.1, 3.2, and 3.6. Retailers can pay the marketplace to advertise their products to consumers. Since the marketplace demands a fee for every single view of the advertisement, retailers want the marketplace to only show the advertisement to those consumers that are likely to buy the product. One possible way to choose suitable consumers is to consider only those who buy a variety of products from the same or a similar product group as the advertised product and who are thus more likely to try new products that are similar to the advertised one. At the same time, the money spent by the chosen consumers on the product group should be above average.

In the previous examples, we have already seen a formula $\varphi_{\text{spending}}(c)$ that defines consumers who have spent at least as much as the average consumer on the product group. The formula depends on a formula $\varphi_{\text{group}}(p)$ that defines a certain group of products based on the structure of their transactions. Due to the connection between graph neural networks and the Weisfeiler-Leman algorithm described in [16], we may assume that there is a formula in $\text{FO}[\sigma]$ that at least roughly approximates such a product group. Likewise, we might assume that there is a formula $\varphi_{\text{variety}}(c)$ in $\text{FO}[\sigma]$ that defines consumers with a wide variety of products bought from a specific product group. However, it is a non-trivial task to design such formulas by hand. It is even not clear whether there exist better rules for finding suitable consumers. Meanwhile, we can easily show the advertisement to consumers and then check whether they buy the product. Thus, we can generate a list with positive and negative examples of consumers. Since the proposed rule can be defined in $\text{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ as $\varphi_{\text{advertise}}(c) := (\varphi_{\text{variety}}(c) \wedge \varphi_{\text{spending}}(c))$, we can use one of the learning algorithms from Theorem 6.1 to find good definitions for $\varphi_{\text{variety}}(c)$ and $\varphi_{\text{group}}(p)$ or to learn an even better definition for $\varphi_{\text{advertise}}(c)$ in $\text{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ from examples.

We believe that our results can be generalised to an extension of FOWA_1 where constructions of the form $\mathsf{P}(t_1, \ldots, t_m)$ are not restricted to the case that $|V| = 1$ for $V := \text{free}(t_1) \cup \cdots \cup \text{free}(t_m)$, but may also be used in a guarded setting of the form $\big(\mathsf{P}(t_1, \ldots, t_m) \wedge \bigwedge_{v,w \in V} \text{dist}(v, w) \leqslant r\big)$. It would also be interesting to study non-Boolean classification problems, where classifiers are described by $\mathbb{S}$-terms defined in a suitable fragment of FOWA. We plan to do this in future work.

References

1 Dana Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, 1987. doi:10.1007/BF00116828.

2 Haim Gaifman. On local and non-local properties. In Jacques Stern, editor, *Proceedings of the Herbrand Symposium*, volume 107 of *Studies in Logic and the Foundations of Mathematics*, pages 105–135. North-Holland, 1982. doi:10.1016/S0049-237X(08)71879-2.

3 Emilie Grienenberger and Martin Ritzert. Learning definable hypotheses on trees. In *22nd International Conference on Database Theory, ICDT 2019, March 26-28, 2019, Lisbon, Portugal*, pages 24:1–24:18, 2019. doi:10.4230/LIPIcs.ICDT.2019.24.

4 Martin Grohe. Logic, graphs, and algorithms. In *Logic and Automata: History and Perspectives [in Honor of Wolfgang Thomas]*, volume 2 of *Texts in Logic and Games*, pages 357–422. Amsterdam University Press, 2008.

5 Martin Grohe. word2vec, node2vec, graph2vec, x2vec: Towards a theory of vector embeddings of structured data. In Dan Suciu, Yufei Tao, and Zhewei Wei, editors, *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS*

2020, Portland, OR, USA, June 14-19, 2020, pages 1–16. ACM, 2020. doi:10.1145/3375395.3387641.

6 Martin Grohe, Christof Löding, and Martin Ritzert. Learning MSO-definable hypotheses on strings. In *International Conference on Algorithmic Learning Theory, ALT 2017, 15-17 October 2017, Kyoto University, Kyoto, Japan*, pages 434–451, 2017. URL: http://proceedings.mlr.press/v76/grohe17a.html.

7 Martin Grohe and Martin Ritzert. Learning first-order definable concepts over structures of small degree. In *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavik, Iceland, June 20-23, 2017*, pages 1–12, 2017. doi:10.1109/LICS.2017.8005080.

8 Martin Grohe and Nicole Schweikardt. First-order query evaluation with cardinality conditions. In *Proceedings of the 37th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, Houston, TX, USA, June 10-15, 2018*, pages 253–266, 2018. doi:10.1145/3196959.3196970.

9 Martin Grohe and György Turán. Learnability and definability in trees and similar structures. *Theory Comput. Syst.*, 37(1):193–220, 2004. doi:10.1007/s00224-003-1112-8.

10 Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 855–864, 2016. doi:10.1145/2939672.2939754.

11 David Haussler. Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Inf. Comput.*, 100(1):78–150, 1992. doi:10.1016/0890-5401(92)90010-D.

12 Michael J. Kearns and Umesh V. Vazirani. *An Introduction to Computational Learning Theory.* MIT Press, 1994. URL: https://mitpress.mit.edu/books/introduction-computational-learning-theory.

13 Dietrich Kuske and Nicole Schweikardt. First-order logic with counting. In *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavik, Iceland, June 20-23, 2017*, pages 1–12, 2017. doi:10.1109/LICS.2017.8005133.

14 Dietrich Kuske and Nicole Schweikardt. Gaifman normal forms for counting extensions of first-order logic. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, pages 133:1–133:14, 2018. doi:10.4230/LIPIcs.ICALP.2018.133.

15 Leonid Libkin. *Elements of Finite Model Theory.* Texts in Theoretical Computer Science. An EATCS Series. Springer, 2004. doi:10.1007/978-3-662-07003-1.

16 Christopher Morris, Martin Ritzert, Matthias Fey, William L. Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and Leman go neural: Higher-order graph neural networks. In *The 33rd AAAI Conference on Artificial Intelligence, AAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 4602–4609, 2019. doi:10.1609/aaai.v33i01.33014602.

17 Shimei Pan and Tao Ding. Social media-based user embedding: A literature review. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 6318–6324, 2019. doi:10.24963/ijcai.2019/881.

18 Maximilian Schleich, Dan Olteanu, Mahmoud Abo Khamis, Hung Q. Ngo, and XuanLong Nguyen. Learning models over relational data: A brief tutorial. In Nahla Ben Amor, Benjamin Quost, and Martin Theobald, editors, *Scalable Uncertainty Management - 13th International Conference, SUM 2019, Compiègne, France, December 16-18, 2019, Proceedings*, volume 11940 of *Lecture Notes in Computer Science*, pages 423–432. Springer, 2019. doi:10.1007/978-3-030-35514-2_32.

19 Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms.* Cambridge University Press, New York, NY, USA, 2014.

20 Szymon Toruńczyk. Aggregate queries on sparse databases. In Dan Suciu, Yufei Tao, and Zhewei Wei, editors, *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2020, Portland, OR, USA, June 14-19, 2020*, pages 427–443. ACM, 2020. doi:10.1145/3375395.3387660.

21 Leslie G. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, 1984. doi:10.1145/1968.1972.

22 Steffen van Bergerem. Learning concepts definable in first-order logic with counting. In *34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2019, Vancouver, BC, Canada, June 24-27, 2019*, pages 1–13, 2019. doi:10.1109/LICS.2019.8785811.

23 Steffen van Bergerem and Nicole Schweikardt. Learning concepts described by weight aggregation logic. *CoRR*, abs/2009.10574, 2020. arXiv:2009.10574.

24 Vladimir Vapnik. Principles of risk minimization for learning theory. In *Advances in Neural Information Processing Systems 4, [NIPS Conference, Denver, Colorado, USA, December 2-5, 1991]*, pages 831–838, 1991. URL: http://papers.nips.cc/paper/506-principles-of-risk-minimization-for-learning-theory.