Web-Mining Agents

Prof. Dr. Ralf Möller

Dr. Özgür Özçep

Universität zu Lübeck

Institut für Informationssysteme

Tanya Braun (Lab Class)

Learning FOL Definable Concepts

slides prepared by Özgür Özçep

Literature

M. Grohe & Martin Ritzert: Learning first-order definable concepts over structures of small degree, arxiv: 1701.05487v1, 19 Jan 2017

(https://arxiv.org/abs/1701.05487)

Declarative Approach

- Grohe, Ritzert 17) is a Logic-based framework for learning
  - Based on background knowledge given by a structure in a logical sense
- Context: Logical and relational Learning
  - Inductive Logic Programming
  - (Statistical) relational Learning
  - Mining and Learning in Graphs
  - ...
- Overview: see IJCAI 09 tutorial of Luc de Raedt
  - (*www.cs.kuleuven.be/~lucdr/ijcai09.pdf*)
  - Book: Logical and relational Learning, Springer, 2008

Scenario & Terminology

- U = instance space
- C* = target concept C: U → {1,0}
- H = hypothesis regarding concept
- Aim: Reduce prediction (true) error of H
- Supervised scenario:
  - Training sequence T consists of labelled examples:
    (ui, C*(ui))

Approach of (Grohe/Ritzert 17)

Note that „model" here denotes a formula, not – in the logical sense - a structure making a formula true

- Background structure B with domain D = Dom(B)
- Parameterized model class = hypotheses space = set of φ(x;y) over logic L where
  - x = instance variable vector (length k)
  - y = parameter variable vector (length l)
  - U = instance space = D^k
  - For each parameter instance v of y one has a hypothesis or an L-definable model
  - $[[\varphi(x,v)]]^B(u) := \begin{cases} 1 & \text{if } B \vDash \varphi(u;v) \\ 0 & \text{otherwise} \end{cases}$

Example

- B = {E,R}-structure
  (directed graph with red-coloring)
- Input sequence: (a,0), (b,1), (g,0), (k,1)

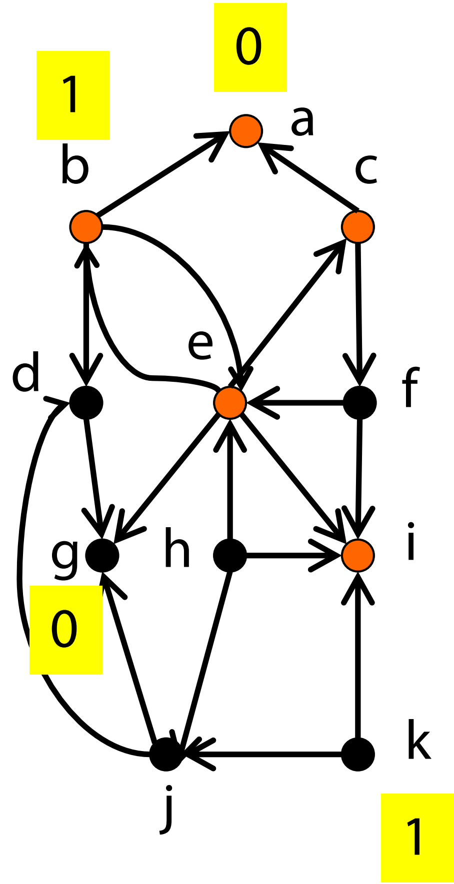
- Model formula

  $\varphi(x;y_1,y_2) :=$

  $(R(x) \vee x = y_1 \vee E(x,y_1)) \;\&$

  $\neg \exists z\, (E(y_2,z) \;\&\; E(z,x))$

- Consistent hypothesis: $[[\varphi(x;j,e)]]^B$



Two approaches

- Parameter learning (see example above)
  - Model (parameterized formula φ) given
  - Have to identify paramters

- Model learning
  - Have to guess both: formula and parameters
  - Compare this with the approach of Gaussian processes
  - Results of (Grohe, Ritzert 17) mainly for this mode

Lit: Rasmussen, Williams: Gaussian Processes for Machine Learning, MIT Press, 2005

Research program

- Identify logics L suitable for expressing ML models (φ) and study their algorithmic learnability (efficient algorithms, lower bounds ...)

- "Descriptive Complexity for ML"

How to access background structure?

1. B not part of the input
   - B may be infinite
   - Algorithm can store elements of B in memory cell and use operations of B (compare register model)
   - Access to B may be local (see below)

Neighborhood defined w.r.t. Gaifman graph of B

2. B part of the input with local access
   - B is finite (but possible very large)
   - Only „local access" of the following types granted
     - Relation query: Is $(u_1, ..., u_n)$ in R? → constant time
     - Neighboorhood query: Give all neighbours for a u in Dom(B) → time proportional to size of answer set

„Low-degree“ background stucture

- Results of Grohe/Ritzert for low-degree structures B
- G(B) = Gaifman graph of B = graph with edge relation E where E(a,b) iff a,b contained in some relation of B
- Degree(B) = graph theoretical degree of G(B)
- Consider structures of bounded degree

Terminology

- k-ary learning problem for background structure B
  - Instance space: $D = \text{Dom}(B)$
  - Aim: Learn target concept $C^*: D^k \rightarrow \{0,1\}$

    with small error

  Here: Space C of target concepts induced by FOL formulae

  $C = \{ [[\varphi(x;v)]]^B \mid \varphi(x;y)$ an FOL formula with $\text{qrank} \leq q$, $x =$ k-vector of variables and $y =$ l-vector of parameters $\}$

- Note: instance-space dimension k, parameter dimension l and quantifier rank q fixed beforehand!

Terminology

Learning algorithm LA for k-ary learning problem over B

- Input
  - Sequence T of form (u,C*(u)) for u in D^k
  - Local access to B
- Output
  - Formula φ(x;y) and parameter v in D^k representing hypothesis
    $H = [[\varphi(x;v)]]^B : D^k \rightarrow \{0,1\}$
  - Reject, if no hypothesis found consistent with input

Hypotheses of LA evaluable in time t iff there is an algorithm s.t. given φ(x;y) and v it outputs for every u $[[\varphi(x;v)]]^B(u)$ in time t

Main Theorem (Grohe

If degree d and t polylogarithmic in n (sublinear!!), then so are runtime and evaluation time

Theorem 1.1 Let k,l,q in Nat. There is q* in Nat and a learning algorithm LA for the k-ary learning problem over finite B s.t.

1. If LA returns a hypothesis H, then H is of form $[[\varphi^*(x;v^*)]]^B$ for FOL formula $\varphi^*(x;v)$ with $qrank(\varphi^*) \leq q^*$ and $v \in Dom(B)^l$ and H is consistent with input sequence T

- Model learning problem
- Proof uses locality of FOL

2. If consistent concept induced by some parameters v and formula of $qrank \leq q$ exists, then LA does not reject.

3. Runtime with local access to B is $(\log(n) + d + t)^{O(1)}$
   $n = |D(B)|$, $d = Degree(B)$, $t =$ length of T

4. Output H can be evaluated in time $(\log(n)+ d)^{O(1)}$

Similar complexity results for infinite B and uniform cost measure

PAC Learnability (Grohe, Ritzert 2017)

Corollary

Let d,k,l,q in Nat. For B with degree(B) ≤ d, the class

$C = \{\varphi(x;v)^\wedge B \mid \varphi(x;v)$ is a FOL formula,

$\mathrm{qrank}(\varphi) \leq q, |x|=k, |y| = l, v \in D^l \}$

is PAC-learnable by a LA with local access to B and running in polynomial time in $1/\varepsilon$ and $1/\delta$.

Parameter Learning vs. Model learning

Example

- $\varphi(x;y) = P(y)$
- For any B a $\{P\}$- background structure and any v in $dom(B)$
  - $[[\varphi(x;v)]]^B = 1_x$ if v in P^B and
  - $[[\varphi(x;v)]]^B = 0_x$ otherwise

- Consider B with $P^B = \{v^*\}$
- So target concept $[[\varphi(x;v^*)]]^B = 1_x$ and LA receives only positive examples
- Unless v^* in T, whole B has to be read in worst case.
  - With local access, v^* cannot even be found: $G(B)$ has no edges

Parameter Learning vs. Model learning

Example (continued)

- Intermediate form of incremental change of φ can help: φ_i in determines formula in next step φ_{i+1}

- $\varphi(x;y) = P(y)$
- Consider B with $P^B = \{v^*\}$
- So target concept $[[\varphi(x;v^*)]]^B = 1_x$ and LA receives only positive examples
  - $\varphi_0 = P(y)$
  - φ_0 entails that target concept C is constant
  - After 1 example we know $C = 1_X$ or 0_X (though we do not know the exact parameter)
  - So return $\varphi_1(x;)$ (x=x) or $\varphi_1(x;)$ =not (x=x)

Algorithm idea for Theorem 1.1

- General Idea
  - Instances u of input T induce a substructure of B
  - It is sufficient to consider a „small" neighbourhood N_{2lr^*} of those elements depending on parameter number and degree of B
  - As FOL is local in the model theoretical sense, this local consideration is sufficient
  - Because of this, the hypothesis formulae can be chosen as (syntactically) local formulae
- Implementation
  - Brute force over all possible Gaifman local formula and parameters (in small neighbourhood) and thereby checking consistency

Algorithm for Theorem 1.1

Input: Sequence T, local access operator on B

```
N ← N_{2lr*}(T)
for all v* in N^l do
    for all φ*(x;y) in Φ* do
        consistent ← true
        for all (u,c) in T do
            if ( (N_{r*}(uv*) ⊨ φ*(u;v*) and c = 0)  or
                 (not N_{r*}(uv*) ⊨ φ*(u;v*) and c = 1)   )
            then consistent ← false
        if consistent, then return φ*, v*
reject
```

- Choose small neighbourhoodof elements in input
- Consider all parameter vectors in neighbourhood
- Consider all r*-local FOL formula
- Find (parameter,local formula) pair consistent with input

Algorithm for Theorem 1.1

Input: Sequence T, local access operator on B

1. $N \leftarrow N_{2lr*}(T)$ Choose small neighbourhoodof elements in input
2. **for all** v^* in N^l **do**
3.   **for all** $\varphi^*(x;y)$ in Φ^* **do**
4.   consistent $\leftarrow$ true
5.   **for all** (u,c) in T **do**
6.   **if** ($(N_{r*}(uv^*) \vDash \varphi^*(u;v^*)$ and $c = 0)$ or
7.   (not $N_{r*}(uv^*) \vDash \varphi^*(u;v^*)$ and $c = 1)$)
8.   **then** consistent $\leftarrow$ false
9.   **if** consistent, **then return** φ^*, v^*
10. **reject**

Reminder: Neighbourhood

- B = relational structure with domain $D = Dom(B)$
- $u = (u1, .., uk)$ in D^k

- $N_r(u) = N^B{}_r(u)$ = all elements in D with distance atmost r to one of the elements u_i within G(B)

With abuse of notation

- $N_r(u)$ = r neighbourhood of u in structure B

  = sub-structure of B induced by all elements in

  maximal distance r to one of the u_i

- $N_r(T) = \cup_{1 \leq i \leq t} N_r(u_i)$ for input sequence

  $T = (u_1, C(u_1), (u_2, C(u_2), \ldots, (u_t, C(u_t))$

Algorithm for Theorem 1.1

Input: Sequence T, local access operator on B

Neighbourhood can be constructed with local access only

1. $N \leftarrow N_{2lr^*}(T)$
2. **for all** v^* in N^l **do**
3. **for all** $\varphi^*(x;y)$ in Φ^* **do**
4. consistent $\leftarrow$ true
5. **for all** (u,c) in T **do**
6. **if** ($(N_{r^*}(uv^*) \models \varphi^*(u;v^*)$ and $c = 0)$ or
7. (not $N_{r^*}(uv^*) \models \varphi^*(u;v^*)$ and $c = 1)$)
8. **then** consistent $\leftarrow$ false
9. **if** consistent, **then return** φ^*, v^*
10. **reject**

Algorithm for Theorem 1.1

Input: Sequence T, local access operator on B

$N \leftarrow N_{2 l r^*}(T)$

for all v^* in N^l **do**

  for all $\varphi^*(x;y)$ in Φ^* **do**

    consistent ← true

    for all (u,c) in T **do**

      if (($N_{r^*}(uv^*) \vDash \varphi^*(u;v^*)$ and $c = 0$) or

        (not $N_{r^*}(uv^*) \vDash \varphi^*(u;v^*)$ and $c = 1$))

      then consistent ← false

    if consistent, **then return** φ^*, v^*

reject

Consider all r*-local FOL formula

Reminder: Locality

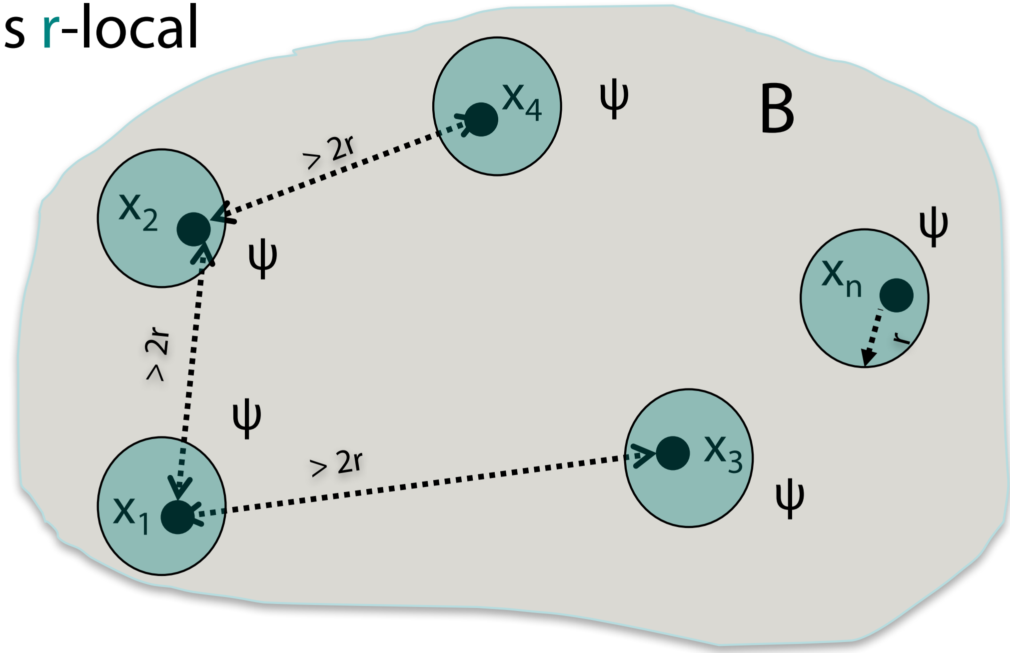
- $\psi(x)$ is r-local iff for all structures B and tuples u

  $$B \vDash \psi(u) \quad \text{iff} \quad N_r(u) \vDash \psi(u)$$

- $\delta_{>r}(x,y)$ = distance of x and y is larger than r (FOL formula)
- $\delta_{<=r}(x,y)$ = distance of x and y smaller than r
- Basic local sentence of radius r

  $$\exists x_1,\ldots, \exists x_k \left(\bigwedge_{1\leq i<j\leq k} \delta_{>2r}(x_i,x_j) \;\&\; \bigwedge_{1\leq i\leq k} \psi(x_i) \right)$$

  where $\psi(x_i)$ is r-local



Reminder: Locality

Theorem (Gaifman): Every FOL formula is equivalent to a boolean combination of basic local sentences and local formulae.

Syntactic notion of locality

- R-relativisation $\varphi_{[<=r]}(x_1, ..,x_k)$: quantifiers of φ relativised to elements in maximally r distance to one of the x_i

  e.g., $\exists y\ \psi(x_1, ..,x_k)$ becomes

  $\exists y\ (\bigvee_{1\leq i\leq k} \delta_{\leq r}(y,x_i)\ \&\ \psi(x_1, ..,x_k))$
- $\varphi(x)$ is syntacically r-local iff it is a r-relativisation

- $\varphi(x)$ is a syntactically basic local sentence if it a basic local sentence $\exists x_1,..., \exists x_k\ (\bigwedge_{1\leq i<j\leq k}\delta_{>2r}(x_i,x_j)\ \&\ \bigwedge_{1\leq i\leq k}\psi(x_i))$ with ψ syntactically r-local

Gaifman normal form

- φ is in Gaifman normal form (GNF) if it is a boolean combination of syntactically basic local sentences and syntactically local sentences.
- Locality radius of GNF φ = least r such that all basic local sentences r-local and all local formulae have radius smaller than r

Corollary

Every FOL formula with at most $k+l$ variables is equivalent to a GNF formula with locality radius r^* such that the quantifier rank of each contained syntactically local formula is smaller than q^*.

- Φ^* := syntactically r-local formula $\varphi(x;y)$ of quantifier rank at most q^*

Algorithm for Theorem 1.1

Input: Sequence T, local access operator on B

$N \leftarrow N_{2l r^*}(T)$

for all v^* in N^l **do**

  for all $\varphi^*(x;y)$ in Φ^* **do**

    $consistent \leftarrow true$

    for all (u,c) in T **do**

      if (($N_{r^*}(uv^*) \models \varphi^*(u;v^*)$ and $c = 0$) or

        (not $N_{r^*}(uv^*) \models \varphi^*(u;v^*)$ and $c = 1$))

      then $consistent \leftarrow false$

    if consistent, **then return** φ^*, v^*

reject

Consider all r*-local FOL formula

Main Theorem (Grohe, Ritzert 17)

Theorem 1.1 Let k,l,q in Nat. There is q* in Nat and a learning algorithm LA for the k-ary learning problem over finite B s.t.

1. If LA returns a hypothesis H, then H is of form $[[\varphi^*(x;v^*)]]^B$ for FOL formula $\varphi^*(x;v)$ with $qrank(\varphi^*) \leq q^*$ and $v \in Dom(B)^l$ and H is consistent with input sequence T
2. If consistent concept induced by some parameters v and formula of $qrank \leq q$ exists, then LA does not reject.
3. Runtime with local access to B is $(\log(n) + d + t)^{O(1)}$
   $N = |D(B)|$, $d = Degree(B)$, $t =$ length of T
4. Output H can be evaluated in time $(\log(n)+ d)^{O(1)}$

Proof hints for Theorem 1.1

First part of Thm 1.1 holds because φ* in LA are r-local:

$$N_r(uv^*) \vDash \varphi^*(u,v) \text{ iff } B \vDash \varphi^*(u,v^*)$$

Main Theorem (Grohe, Ritzert 2017)

Theorem 1.1 Let k,l,q in Nat. There is q* in Nat and a learning algorithm LA for the k-ary learning problem over finite B s.t.

1. If LA returns a hypothesis H, then H is of form $[[\varphi^*(x;v^*)]]^B$ for FOL formula $\varphi^*(x;v)$ with $\mathrm{qrank}(\varphi^*) \leq q^*$ and $v \in \mathrm{Dom}(B)^l$ and H is consistent with input sequence T

2. If consistent concept induced by some parameters v and formula of $\mathrm{qrank} \leq q$ exists, then LA does not reject.

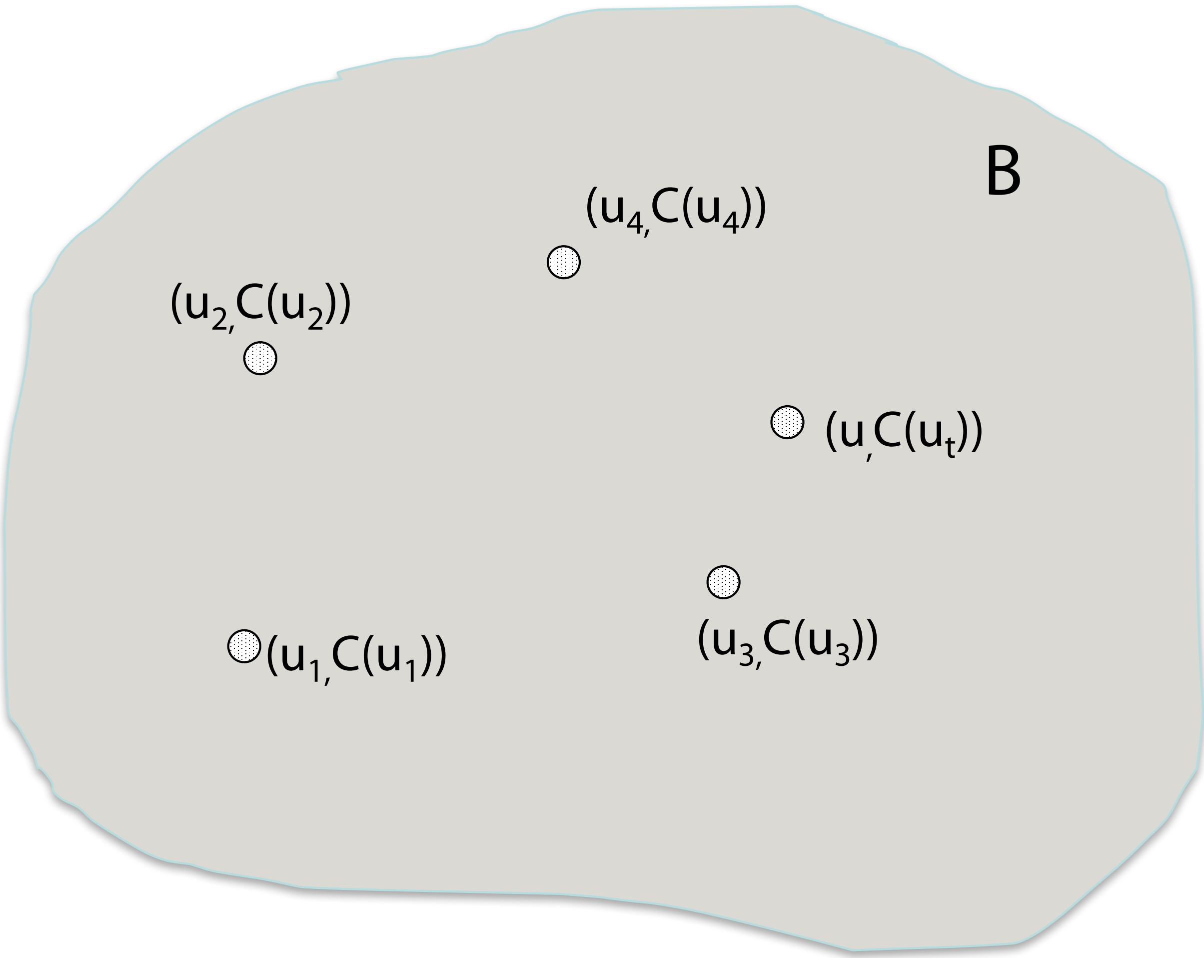
3. Runtime with local access to B is $(\log(n) + d + t)^{O(1)}$
$N = |D(B)|$, $d = \mathrm{Degree}(B)$, $t =$ length of T

4. Output H can be evaluated in time $(\log(n) + d)^{O(1)}$

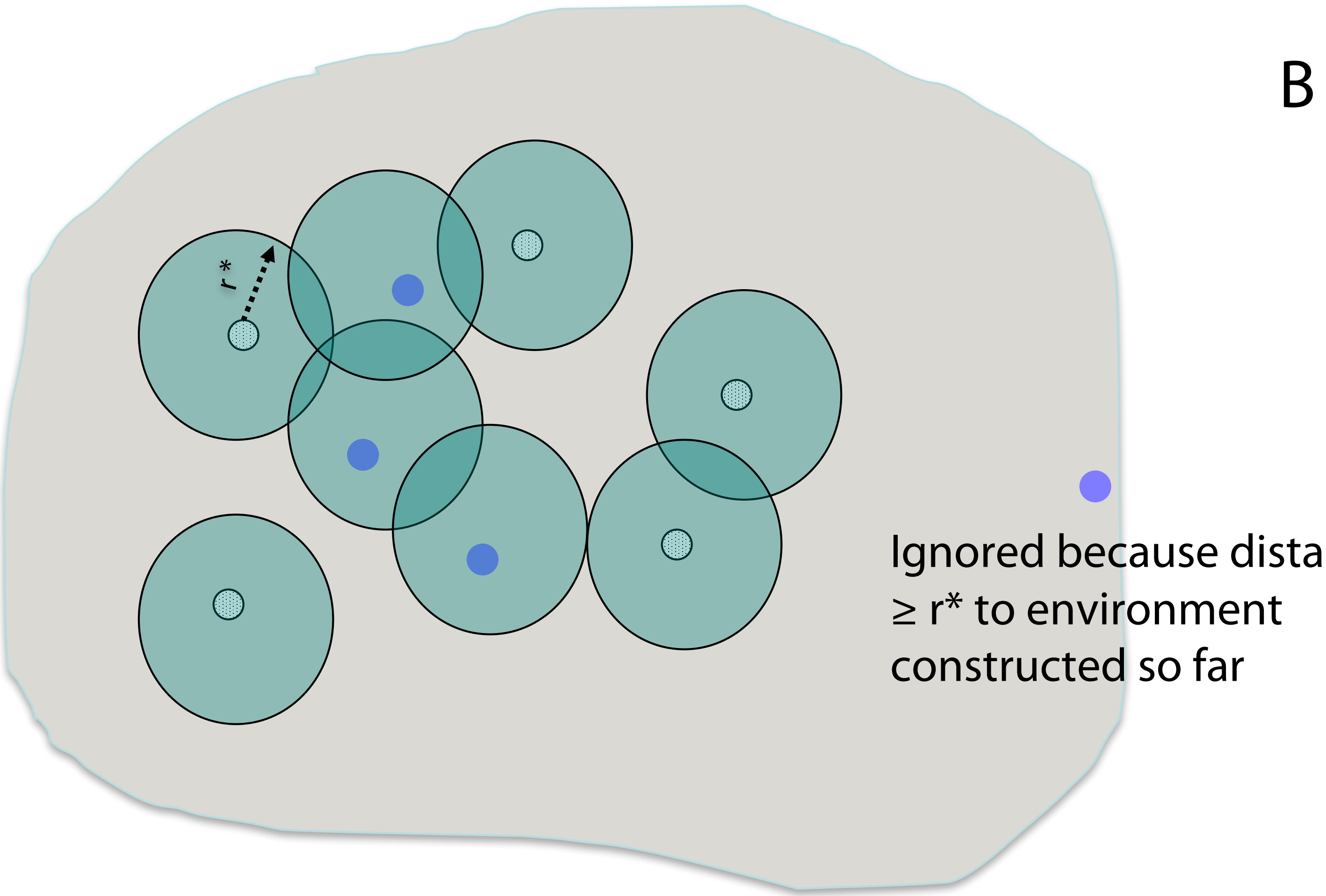
Proof hints on Theorem 1.1 (part 2)

Is true target captured by syntacically local φ^*?

- Assume target concept C given by formula φ and parameter vector v
- Can choose subset v' of paremeters v of true target concept in some local environment (and then possibly pad them to v*)
- Instances together with v describable by formula (their local type)


B
(u4,C(u4))
(u2,C(u2))
(u,C(ut))
(u1,C(u1))
(u3,C(u3))

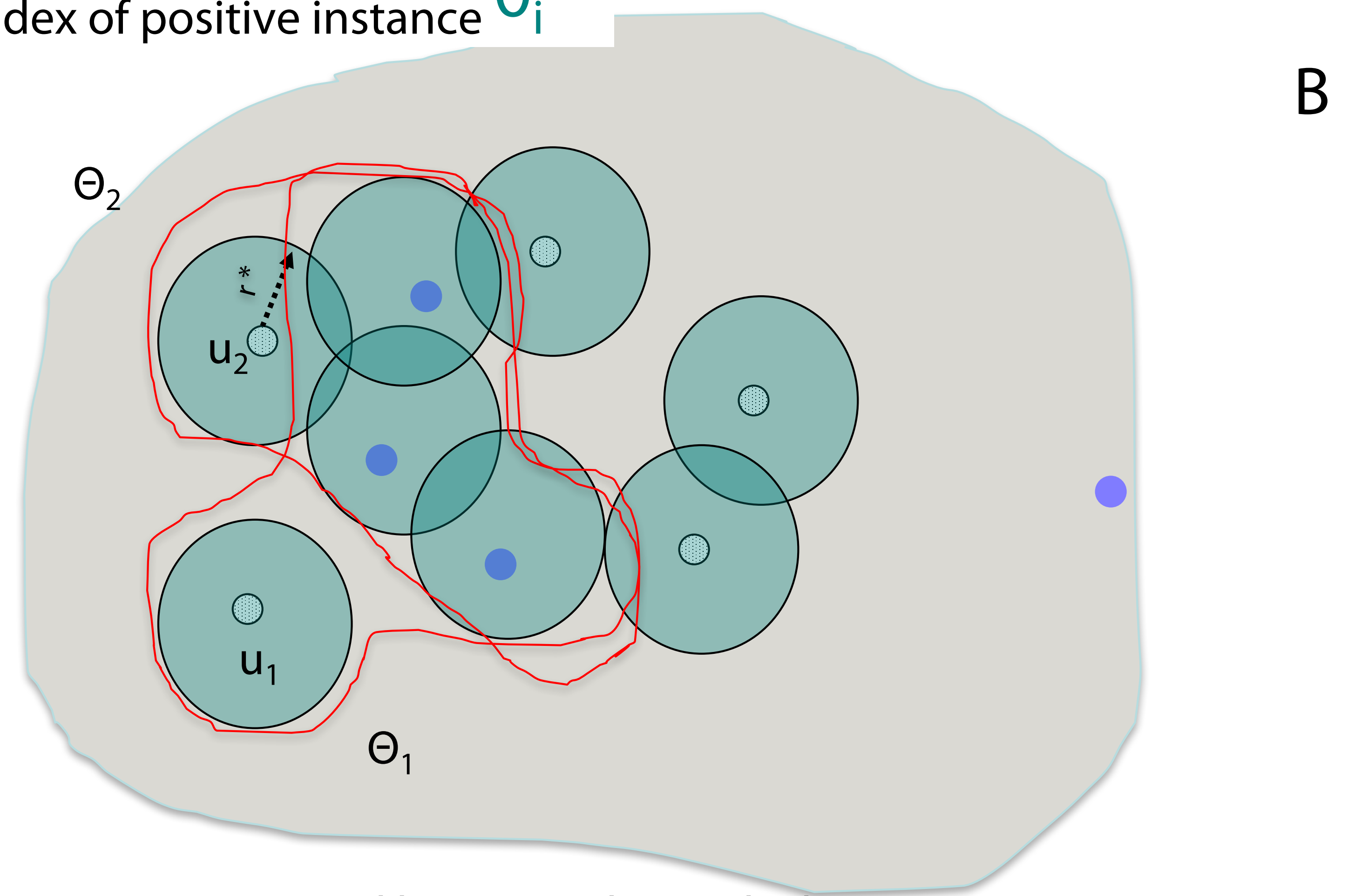
Finding the right parameters



Ignored because distance ≥ r* to environment constructed so far

Chosen parameter components can be padded to v*
In order to produce consistent hypothesis

$\varphi^* = \bigvee_i$ index of positive instance θ_i



Each substructure generated by u_i together with chosen parameter vector v*
describable by a syntactically local formula θ_i (its local type)

$\varphi^* = \bigvee_{i \text{ index of positive instance}} \theta_i$

With Feferman-Vaught style composition lemma can ensure that identical types lead to same categorization

Proof hints on Thm 1

- Ad 3. :
  - $|N_{r*}(uv*)| <= (k+l)2d^{r*}$
  - Thus, $N_{r*}(uv*)$ representable in size $O((k+l)d^{r*}\log(n))$
  - Considering k,l and r* as constants gives
  - $N_{r*}(uv*)$ representable in size $(d + \log(n))^{O(1)}$
  - With locall access need time polynomial in representation size of $N_{r*}(uv*)$ to check wether φ* true in $N_{r*}(uv*)$ hence:
  - Running time consistency checks: $t\,(d + \log(n))^{O(1)}$
  - As $|N| <= 2tkd^{2lr*} = (t+1)^{O(1)}$, hence outer loops add factor $(t+d)^{O(1)}$ and so overall runtime $(\log(n) + d + t)^{O(1)}$
- Ad 4.: As φ* local only check on Nr*(u,v) required.

Result with uniform cost (Grohe, Ritzert 17)

Theorem 4.3 Let k,l,q in Nat. There is q* in Nat and a learning algorithm LA for the k-ary learning problem over (possibly infinite) B s.t.

1. If LA returns an H, then H is of form $[[\varphi^*(x;v^*)]]^B$ for FOL formula $\varphi^*(x;v)$ with $\mathrm{qrank}(\varphi^*) \leq q^*$ and v* in $\mathrm{Dom}(B)^k$ and H is consistent with input sequence T
2. If consistent hypothesis for some parameters v and formula of $\mathrm{qrank} \leq q$ and exists, then LA does not reject.
3. Runtime in uniform cost is = $(d + t)^{O(1)}$ with local access to B
   d = Degree(B), t = length of T
4. Outputted H can b

Uniform cost model as in register machine model:
Arbitrarily large elements in register (constant space)

Generalisation: Minimize error

- Do not require consistent hypothesis but minimal training error of hypothesis H
- $err_T(H) = 1/t \,|\{i \text{ in } \{1, ..., t\} \mid H(u_i) \neq c_i \}|$

Minimizing error

Theorem 4.4 Let k,l,q in Nat. There is q* in Nat and a learning algorithm LA for the k-ary learning problem over finite B s.t.

1. If LA returns an H, then H is of form $[[\varphi^*(x;v^*)]]^B$ for r*-local FOL formula $\varphi^*(x;v)$ with $qrank(\varphi^*) \leq q^*$ and v* in $Dom(B)^k$ and H is consistent with input sequence T

2. If concept induced by some parameters v and formula $\varphi(x;y)$ of $qrank \leq q$ exists with $err_T([[\varphi(x;v)]]) \leq \varepsilon$, then $err_T(H) \leq \varepsilon$ for the H returend by LA on T.

3. Runtime with local access to B = $(\log(n) + d + t)^{O(1)}$
   $N = |D(B)|$, $d = Degree(B)$, t = length of T

4. Outputted H can be evaluated in time $(\log(n)+ d)^{O(1)}$

Algorithm for Theorem 4.4

Input: Sequence T, local access operator on B

```
N ← N_{2lr*}(T)
minerr ← t+1
for all v* in N^l do
   for all φ*(x;y) in Φ* do
      err ← 0
       for all (u,c) in T do
         if ( (N_{r*}(uv*) ⊨ φ*(u;v*) and c = 0)   or
              (not N_{r*}(uv*) ⊨ φ*(u;v*) and c = 1)   )
         then err ← err +1
       if err < minerr, then
            minerr ← err
            φ_min ← φ*
            v_min ← v*
return φ_min, v_min
```

Proof of correctness by reduction to Thm1:
As error ≤ ε, there is (1-ε)*t elements for which target conept
consistent. Now apply Thm 1.1.

PAC learning reminder

- D = instance space   C = target space   H = hypothesis space
- t = function describing number of training samples required by learning algorithm
- C* in C , H in H, P probability distribution on D

- True error: $Err_{P,C^*}(H) = Pr_{x \simeq P}(H(x) \neq C^*(x))$

- Learning algorithm LA is a (D,C,H,t)-PAC-learning algorithm iff
  - For all probability distributions P on D,
  - For all target concepts C* in C
  - For all input sequences T
  - and for all ε,δ >0

  LA generates hypothesis H= H(T,ε, δ) such that

$$Pr_{T \simeq P}(\, errP,C(H) \leq \varepsilon) \geq 1 - \delta$$

PAC learning (Grohe, Ritzert 2017)

Theorem 4.3 Let k,l,q in Nat and
C = target concepts induced by FOL formula φ(x;y) of qrank <= q
with k-vector of instance variables x and l-vector of paramters y

There are q*,r*,s* in Nat and a learning algorithm LA for the k-ary learning problem over finite B s.t.

1. Setting
   - H = hypothesis space induced by r*-local formula w/ qrank ≤ q*
   - $t(n,\varepsilon,\delta) = s^* \log(n/\delta)/\varepsilon$ where $n = |dom(B)|$

   makes LA a $(Dom(B)^k, C, H, t)$-PAC learning algorithm

2. Runtime is = $(\log(n) + degree(B) + 1/\varepsilon + \log(1/\delta))^{O(1)}$
   with local access to B

VC dimension and Degree

- For infinite B one can use the VC dimension
- One knows that the class of FOL concepts on small degree sturctures have a finite VC dimension.

Theorem (Grohe, Turan 04)

Let k,l,q in Nat. For

$\underline{C}$ = target concepts induced by FOL formula φ(x;y) of qrank <= q with k-vector of instance variables x and l-vector of paramters y on structure with finite degree VC($\underline{C}$) is finite.

- Using known bound on samples for concept classes with finite VC dimension one gets PAC learnability for structures with low degrees.

PAC Learning (Grohe, Ritzert 2017)

Theorem 4.3 Let k,l,q in Nat and
C = {concepts induced by FOL formula φ(x;v) of qrank ≤ q.}
There are q*,r*,s* in Nat and a learning algorithm LA for the k-ary learning problem over background structure B with degree d s.t.

1. setting
   - H = { hypotheses induced by r*-local formula with qrank ≤ q* }
   - t(ε,δ) = s* log(n/d)/ε where n = |dom(B)|

   makes LA a $(\mathrm{Dom}(B)^k, C, H, t)$-PAC learning algorithm

2. Runtime is = $(1/\varepsilon + \log(1/\delta))^{O(1)}$ under uniform cost measure and with local access to B only

Discussion

- Interesting framework for ML learning algorithms
- Further generalizations and topics in this framework
  - Further logics
  - Different learning aims
  - Intermediate form of learning and online aspects
  - ...
- Criticisms/Further Research
  - Number of parameters has to be known in advance
    - Though model learning, it depends on this number
  - Dependance on B may cause trouble (?)
    - How „big“ must B be
    - How random mus B be?