Computational Learning in the Limit
Lecture 2: The Inconsistency Phenomenon

Timo Kötzing
MPI for Computer Science

October 8, 2009

Closure Under Union?

$$\mathcal{S}_0 = \{g \in \mathcal{R} \mid \forall^\infty x : g(x) = 0\};$$
$$\mathcal{S}_1 = \{g \in \mathcal{R} \mid \varphi_{g(0)} = g\}.$$

- $\mathcal{S}_0$ is uniformly computable.
- $\mathcal{S}_1$ is not uniformly computable.
- Both $\mathcal{S}_0$ and $\mathcal{S}_1$ are **GEx**-learnable.
- What about $\mathcal{S}_0 \cup \mathcal{S}_1$? Not learnable.

Closure Under Union?

$$\begin{aligned} \mathcal{S}_0 &= \{g \in \mathcal{R} \mid \forall^\infty x : g(x) = 0\}; \\ \mathcal{S}_1 &= \{g \in \mathcal{R} \mid \varphi_{g(0)} = g\}. \end{aligned}$$

- $\mathcal{S}_0$ is uniformly computable.
- $\mathcal{S}_1$ is not uniformly computable.
- Both $\mathcal{S}_0$ and $\mathcal{S}_1$ are **GEx**-learnable.
- What about $\mathcal{S}_0 \cup \mathcal{S}_1$? Not learnable.

Closure Under Union?

$$\begin{aligned} \mathcal{S}_0 &= \{g \in \mathcal{R} \mid \forall^\infty x : g(x) = 0\}; \\ \mathcal{S}_1 &= \{g \in \mathcal{R} \mid \varphi_{g(0)} = g\}. \end{aligned}$$

- $\mathcal{S}_0$ is uniformly computable.
- $\mathcal{S}_1$ is not uniformly computable.
- Both $\mathcal{S}_0$ and $\mathcal{S}_1$ are **GEx**-learnable.
- What about $\mathcal{S}_0 \cup \mathcal{S}_1$? Not learnable.

Closure Under Union?

$$\begin{aligned}\mathcal{S}_0 &= \{g \in \mathcal{R} \mid \forall^\infty x : g(x) = 0\};\\ \mathcal{S}_1 &= \{g \in \mathcal{R} \mid \varphi_{g(0)} = g\}.\end{aligned}$$

- $\mathcal{S}_0$ is uniformly computable.
- $\mathcal{S}_1$ is not uniformly computable.
- Both $\mathcal{S}_0$ and $\mathcal{S}_1$ are **GEx**-learnable.
- What about $\mathcal{S}_0 \cup \mathcal{S}_1$? Not learnable.

Closure Under Union?

$$\mathcal{S}_0 = \{g \in \mathcal{R} \mid \forall^\infty x : g(x) = 0\};$$
$$\mathcal{S}_1 = \{g \in \mathcal{R} \mid \varphi_{g(0)} = g\}.$$

- $\mathcal{S}_0$ is uniformly computable.
- $\mathcal{S}_1$ is not uniformly computable.
- Both $\mathcal{S}_0$ and $\mathcal{S}_1$ are **GEx**-learnable.
- What about $\mathcal{S}_0 \cup \mathcal{S}_1$? Not learnable.

Closure Under Union?

$$\mathcal{S}_0 = \{g \in \mathcal{R} \mid \forall^\infty x : g(x) = 0\};$$
$$\mathcal{S}_1 = \{g \in \mathcal{R} \mid \varphi_{g(0)} = g\}.$$

- $\mathcal{S}_0$ is uniformly computable.
- $\mathcal{S}_1$ is not uniformly computable.
- Both $\mathcal{S}_0$ and $\mathcal{S}_1$ are **GEx**-learnable.
- What about $\mathcal{S}_0 \cup \mathcal{S}_1$? Not learnable.

Postdictive Completeness

- A learner h is called postdictively complete iff $\forall g, k$, $h(g[k])$ correctly postdicts $g[k]$ $(= g(0), \ldots, g(k-1))$, i.e., the programs $h(g[k])$ correctly compute g on all inputs $< k$.
- The class of all sets of functions **GEx**-learnable postdictively completely is denoted by **GPcpEx**.
- All uniformly computable sets of functions are **GPcpEx**-learnable (including $\mathcal{S}_0$).
- $\mathcal{S}_1$ is also **GPcpEx**-learnable.
- There are sets of uniformly computable functions (for example **EXPF**) which are not **GPcpEx**-learnable in linear time.
- Is there a set which is **GEx**-learnable, but not **GPcpEx**-learnable (using no runtime restriction)? Yes.

Postdictive Completeness

- A learner h is called postdictively complete iff $\forall g, k$, $h(g[k])$ correctly postdicts $g[k]$ $(= g(0), \ldots, g(k-1))$, i.e., the programs $h(g[k])$ correctly compute g on all inputs $< k$.
- The class of all sets of functions **GEx**-learnable postdictively completely is denoted by **GPcpEx**.
- All uniformly computable sets of functions are **GPcpEx**-learnable (including $\mathcal{S}_0$).
- $\mathcal{S}_1$ is also **GPcpEx**-learnable.
- There are sets of uniformly computable functions (for example **EXPF**) which are not **GPcpEx**-learnable in linear time.
- Is there a set which is **GEx**-learnable, but not **GPcpEx**-learnable (using no runtime restriction)? Yes.

Postdictive Completeness

- A learner h is called postdictively complete iff $\forall g, k$, $h(g[k])$ correctly postdicts $g[k]$ $(= g(0), \ldots, g(k-1))$, i.e., the programs $h(g[k])$ correctly compute g on all inputs $< k$.
- The class of all sets of functions **GEx**-learnable postdictively completely is denoted by **GPcpEx**.
- All uniformly computable sets of functions are **GPcpEx**-learnable (including $\mathcal{S}_0$).
- $\mathcal{S}_1$ is also **GPcpEx**-learnable.
- There are sets of uniformly computable functions (for example **EXPF**) which are not **GPcpEx**-learnable in linear time.
- Is there a set which is **GEx**-learnable, but not **GPcpEx**-learnable (using no runtime restriction)? Yes.

Postdictive Completeness

- A learner h is called postdictively complete iff $\forall g, k$, $h(g[k])$ correctly postdicts $g[k]$ $(= g(0), \ldots, g(k-1))$, i.e., the programs $h(g[k])$ correctly compute g on all inputs $< k$.
- The class of all sets of functions **GEx**-learnable postdictively completely is denoted by **GPcpEx**.
- All uniformly computable sets of functions are **GPcpEx**-learnable (including $\mathcal{S}_0$).
- $\mathcal{S}_1$ is also **GPcpEx**-learnable.
- There are sets of uniformly computable functions (for example **EXPF**) which are not **GPcpEx**-learnable in linear time.
- Is there a set which is **GEx**-learnable, but not **GPcpEx**-learnable (using no runtime restriction)? Yes.

Postdictive Completeness

- A learner h is called postdictively complete iff $\forall g, k,\ h(g[k])$ correctly postdicts $g[k]$ $(= g(0), \ldots, g(k-1))$, i.e., the programs $h(g[k])$ correctly compute g on all inputs $< k$.
- The class of all sets of functions **GEx**-learnable postdictively completely is denoted by **GPcpEx**.
- All uniformly computable sets of functions are **GPcpEx**-learnable (including $\mathcal{S}_0$).
- $\mathcal{S}_1$ is also **GPcpEx**-learnable.
- There are sets of uniformly computable functions (for example **EXPF**) which are not **GPcpEx**-learnable in linear time.
- Is there a set which is **GEx**-learnable, but not **GPcpEx**-learnable (using no runtime restriction)? Yes.

Postdictive Completeness

- A learner h is called postdictively complete iff $\forall g, k,\ h(g[k])$ correctly postdicts $g[k]$ $(= g(0), \ldots, g(k-1))$, i.e., the programs $h(g[k])$ correctly compute g on all inputs $< k$.
- The class of all sets of functions **GEx**-learnable postdictively completely is denoted by **GPcpEx**.
- All uniformly computable sets of functions are **GPcpEx**-learnable (including $\mathcal{S}_0$).
- $\mathcal{S}_1$ is also **GPcpEx**-learnable.
- There are sets of uniformly computable functions (for example **EXPF**) which are not **GPcpEx**-learnable in linear time.
- Is there a set which is **GEx**-learnable, but not **GPcpEx**-learnable (using no runtime restriction)? Yes.

Postdictive Completeness

- A learner h is called postdictively complete iff $\forall g, k$, $h(g[k])$ correctly postdicts $g[k]$ $(= g(0), \ldots, g(k-1))$, i.e., the programs $h(g[k])$ correctly compute g on all inputs $< k$.
- The class of all sets of functions **GEx**-learnable postdictively completely is denoted by **GPcpEx**.
- All uniformly computable sets of functions are **GPcpEx**-learnable (including $\mathcal{S}_0$).
- $\mathcal{S}_1$ is also **GPcpEx**-learnable.
- There are sets of uniformly computable functions (for example **EXPF**) which are not **GPcpEx**-learnable in linear time.
- Is there a set which is **GEx**-learnable, but not **GPcpEx**-learnable (using no runtime restriction)? Yes.

Postdictive Completeness

- A learner h is called postdictively complete iff $\forall g, k$, $h(g[k])$ correctly postdicts $g[k]$ $(= g(0), \ldots, g(k-1))$, i.e., the programs $h(g[k])$ correctly compute g on all inputs $< k$.
- The class of all sets of functions **GEx**-learnable postdictively completely is denoted by **GPcpEx**.
- All uniformly computable sets of functions are **GPcpEx**-learnable (including $\mathcal{S}_0$).
- $\mathcal{S}_1$ is also **GPcpEx**-learnable.
- There are sets of uniformly computable functions (for example **EXPF**) which are not **GPcpEx**-learnable in linear time.
- Is there a set which is **GEx**-learnable, but not **GPcpEx**-learnable (using no runtime restriction)? Yes.

Popperian Learnability

- What if we require each output by the learner to be a program for a total function?
- With other words, what if we only allow learners such that

$$\forall \sigma : \varphi_{h(\sigma)} \in \mathcal{R}?$$

- What can be learned, what cannot be learned? The class of learnable sets of functions is exactly the class of uniformly computable functions.

Popperian Learnability

- What if we require each output by the learner to be a program for a total function?
- With other words, what if we only allow learners such that

$$\forall \sigma : \varphi_{h(\sigma)} \in \mathcal{R}?$$

- What can be learned, what cannot be learned? The class of learnable sets of functions is exactly the class of uniformly computable functions.

Popperian Learnability

- What if we require each output by the learner to be a program for a total function?
- With other words, what if we only allow learners such that

$$\forall \sigma : \varphi_{h(\sigma)} \in \mathcal{R}?$$

- What can be learned, what cannot be learned? The class of learnable sets of functions is exactly the class of uniformly computable functions.

Popperian Learnability

- What if we require each output by the learner to be a program for a total function?
- With other words, what if we only allow learners such that

$$\forall \sigma : \varphi_{h(\sigma)} \in \mathcal{R}?$$

- What can be learned, what cannot be learned? The class of learnable sets of functions is exactly the class of uniformly computable functions.

Popperian Learnability

- What if we require each output by the learner to be a program for a total function?
- With other words, what if we only allow learners such that

$$\forall \sigma : \varphi_{h(\sigma)} \in \mathcal{R}?$$

- What can be learned, what cannot be learned? The class of learnable sets of functions is exactly the class of uniformly computable functions.

Thank You.