Learning with Relaxed Supervision

Jacob Steinhardt  Percy Liang
{jsteinhardt, pliang}@cs.stanford.edu

Intractable Supervision
For weakly-supervised tasks, inference can be intractable:

\[
\text{input } z : \text{What is the largest city in California?}
\]
\[
\text{latent } z : \text{argmax(\lambda x.CITY(x) \wedge \text{LOG}(x, \text{CA}), \lambda x.POPULATION(x))}
\]
\[
\text{output } y : \text{Los Angeles}
\]

Computing \( p(z | x, y) \) requires inverting arbitrary logical forms!
- Still want to exploit likely statistical relationships (CITY and Los Angeles)
- Need a way to relax the supervision so we can learn tractably.
- Want to maintain good statistical properties (asymptotic consistency).

Our Approach

\[
\begin{array}{c|c}
\text{more exact} & \text{more accurate} \\
\hline
\text{less accurate} & \text{tractable region} \\
\hline
\text{intractable region} & \text{learning trajectory} \\
\end{array}
\]

- Start with intractable supervision \( q_0(y | z) \)
- Replace with family of relaxed supervision functions \( q(y | z) \)
- Derive constraints on \((\theta, \beta)\) that ensure tractability of inference
- Optimize likelihood within the tractable region

Intuition:
- Supervision is intractable if too harsh relative to model accuracy.
- Initially need very forgiving supervision, can eventually incorporate full supervision (done adaptively over course of optimization).

The Relaxation
Assume relationship between \( z \) and \( y \) given by constraints \( S_{j, z} \), \( j = 1, \ldots, k \)
(translate into \( y \))

Relaxation based on weighted count of constraint violations:

\[
p_y(y | z) \propto \exp \left( - \sum_{j=1}^k \beta_j (1 - S_j(z, y)) \right)
\]

When \( \beta = 0 \), \( p_y \) is uniform; when \( \beta = \infty \), \( p_y \) is original supervision.

Challenges: normalization constant of \( p_y \); ensuring tractable inference.

Framework
Assumptions:
- \( x \to z \to y \), where \((x, y) \in X \times Y\) is observed and \( z \in Z \) is unobserved.
- Parameterized family \( p_y(z | x) \).
- \( x \to y \) is a known deterministic function \( y = f(z) \).

Hence, letting \( S(z, y) \in [0, 1] \) denote the constraint \( |y = f(z)| \):

\[
p_y(y | x) = \sum_z S(z, y) p_y(z | x).
\]

Goal: decompose \( S \) into smaller components \( S_j \).

- Derive projections \( \pi_j : Y \to Y_j \).
- Projected constraint: \( S(z,y) \implies \pi_j(y) = \pi_j(y) \).
- If \( \pi_1 \times \cdots \times \pi_k \) is one-to-one, then can decompose \( S \) as \( S = \bigwedge_{j=1}^k S_j \).

Example Decompositions
Translation from Unordered Supervision
\[
\begin{array}{c|c|c|c|c|c}
\text{input } x : & a & b & a & a & a \\
\text{latent } z : & c & d & c & d & d \\
\text{output } y : & \{a \to d, b \to z, \ldots\} \\
\end{array}
\]

\[
\text{Supervision: } y = \text{multiset}(z)
\]

Decomposition (y and z match if all counts match):

\[
y = \text{multiset}(z) \iff \bigwedge_{j=1}^{k} \left( \frac{\pi_j(y)}{\sum_{j=1}^{k} \pi_j(y)} \right) = S_j(z, y)
\]

Conjunctive Semantic Parsing
\[
\text{Side information: predicates } Q_1, \ldots, Q_n.
\]
\[
\text{e.g. } Q_1 = \text{[dog]} \text{ is set of all dogs}
\]
\[
\text{input } x : \text{brown dog} \quad \text{(input utterance)}
\]
\[
\text{latent } z : \text{(set of all brown objects, set of all dogs)}
\]
\[
\text{output } y : Q_1 \cap Q_2 \quad \text{(denoting observed as a set)}
\]

For \( z = (Q_1, \ldots, Q_k) \), define the denotation \( [z] = Q_1 \cap \cdots \cap Q_k \).

Decomposition (y and z match if contained in same predicates):

\[
y = [z] \iff \bigwedge_{j=1}^{m} \left( \frac{\pi_j(y)}{\sum_{j=1}^{m} \pi_j(y)} \right) = S_j(z, y)
\]

Theory

Lemma (normalization constant). For any \( z \), the log-normalization constant of \( p_y(y | z) \) is bounded above by

\[
A(\beta) \equiv \sum_{j=1}^{k} \log(1 + (|Y_j| - 1) \exp(-\beta_j / |Y_j|)).
\]

Lemma (asymptotic consistency). Suppose that we use \( A(\beta) \) above as a surrogate normalization constant for \( p_y \). Then, the MLE \( \hat{\beta} \) asymptotically recovers the true model parameters.

Tractability Constraints
Typical expression for gradient (for some features \( f(x, z, y) \)):

\[
\nabla \log p_y(y | x) = \sum_{z \in Z} p_y(z | x) \sum_{y \in Y} \frac{\partial}{\partial y} p_y(y | x, z)
\]

To learn, need to sample \( p_y(y | x, z) \) \( \propto p_y(z | x) \exp(\beta f(z, y)) \) (see (i)).

- For large \( \beta \), this is as intractable as the original supervision.
- Need a way to constrain \( \beta \) to yield tractable inference.
- Inference algorithm: rejection sampling.
- Sample from \( p_y(z | x) \), accept with probability \( p_y(y | z) \).

Constrain expected number of rejections based on computational budget \( \tau \).

\[
\text{minimize } \mathbb{E}_{y \sim p_y} [\text{log}(p_y(y | z))] \quad \text{(C)}
\]
\[
\text{subject to } \mathbb{E}_{y \sim p_y} [\text{Rejections}(x, y)] \leq \tau \quad \text{(C)}
\]

Amazingly, (C) is well-behaved enough to admit an EM-like procedure for constrained optimization! (See paper for full details.)

Experiments
Implemented our relaxed supervision algorithm on both the unordered translation and conjunctive semantic parsing tasks.

Compared a fixed value of relaxation \( \beta \) (FIXED) to optimizing \( \beta \) subject to our tractability constraints (ADAPT).

Our tractability constraints improve efficiency by orders of magnitude while also improving accuracy.

Reproducible experiments on Codalab: \url{worksheets.codalab.org}

The first author was supported by the Henry Foundation and by the NSF.