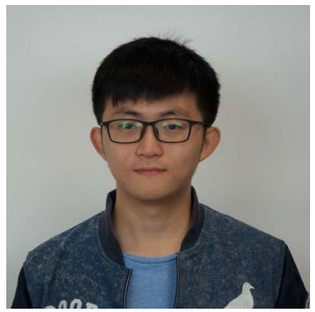
Search-Guided, Lightly-Supervised Training of Structured Prediction Energy Networks

Pedram Rooshenas

Dongxu Zhang



Gopal Sharma

Andrew McCallum

UMassAmherst

Structured Prediction

- We are interested to learn a function $\Phi : \mathbf{X} \rightarrow \mathbf{Y}$
  - X input variables
  - Y output variables

- We can define Φ as $\Phi := \text{argmax}_{\mathbf{y}} P(\mathbf{Y} = \mathbf{y} | \mathbf{x})$
  - For a Gibbs distribution:

$$\Phi := \text{argmin}_{\mathbf{y}} E_{\mathbf{w}}(\mathbf{y}, \mathbf{x})$$

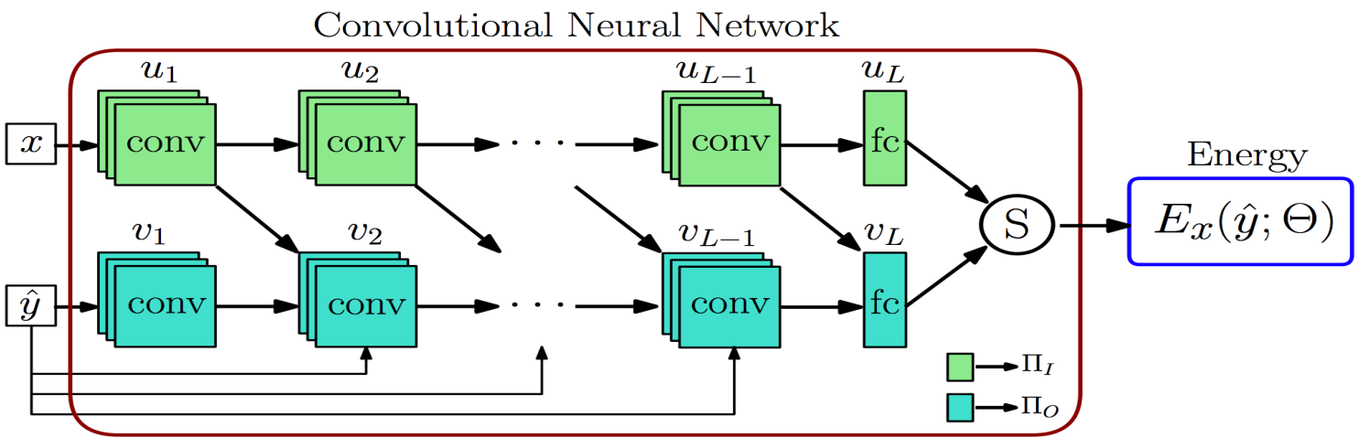
Structured Prediction Energy Networks (SPENs)

- If $E_{\mathbf{w}}(\mathbf{y}, \mathbf{x})$ is parameterized using a differentiable model such as a deep neural network:
  - We can find a local minimum of E using gradient descent

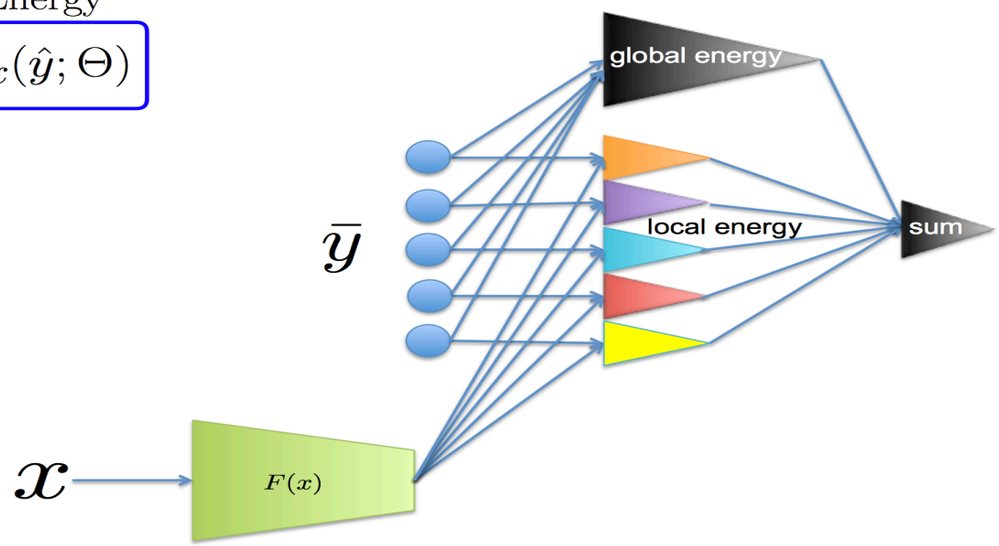
$$\mathbf{y}_{t+1} = \mathcal{P}_{y \in \Delta_L}(\mathbf{y}_t - \eta \frac{\partial}{\partial \mathbf{y}} E_{\mathbf{w}}(\mathbf{y}_t, \mathbf{x}))$$

  - The energy networks express the correlation among input and output variables.
    - Traditionally graphical models are used for representing the correlation among output variables.
      - Inference is intractable for most of expressive graphical models

Energy Models



[picture from Altinel (2018)]



[picture from Belanger (2016)]

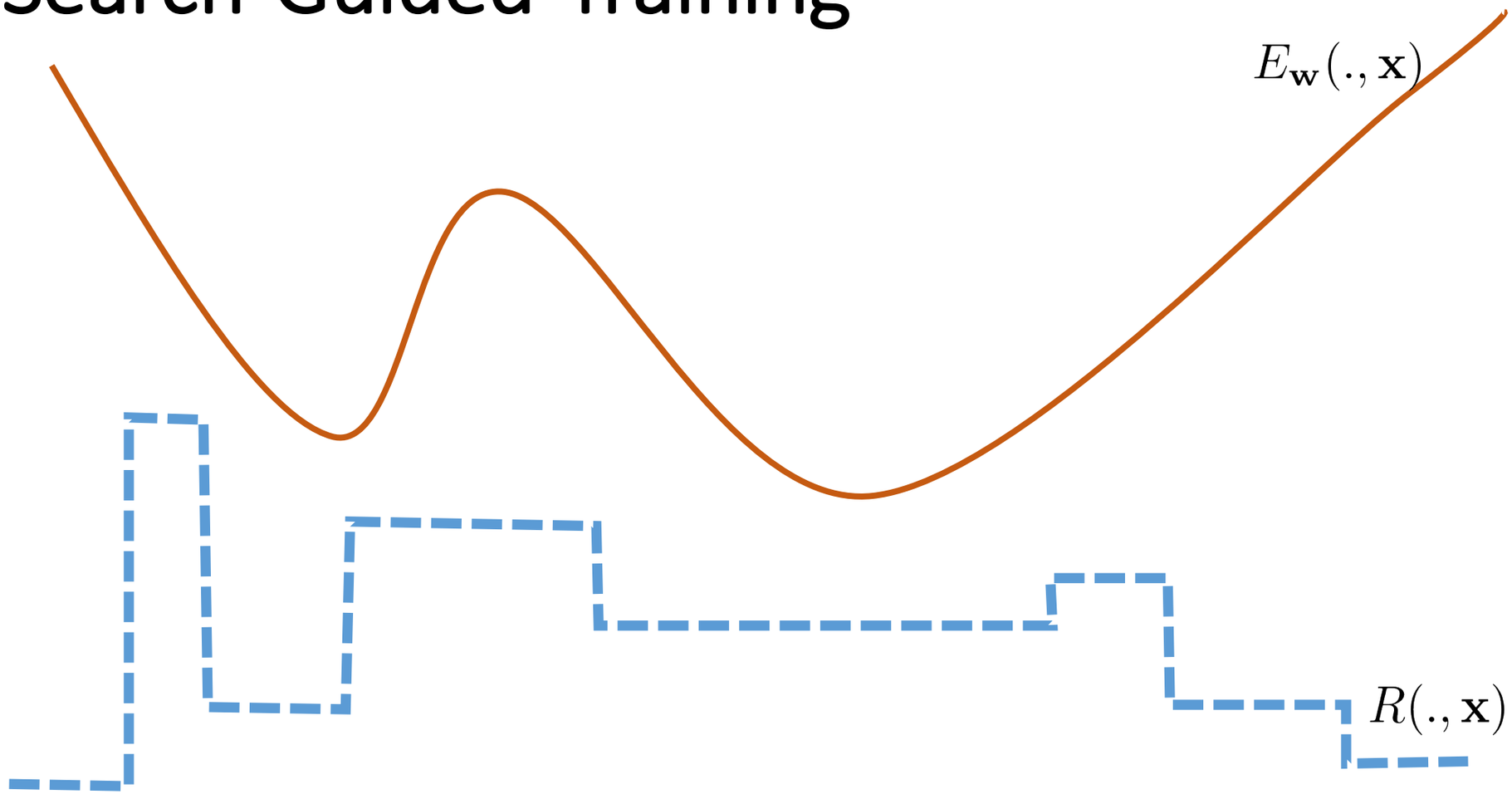
Training SPENs

- Structural SVM (Belanger and McCallum, 2016)
- End-to-End (Belanger et al., 2017)
- Value-based training (Gygli et al. 2017)
- Inference Network (Lifu Tu and Kevin Gimpel, 2018)
- Rank-Based Training (Rooshenas et al., 2018)

Indirect Supervision

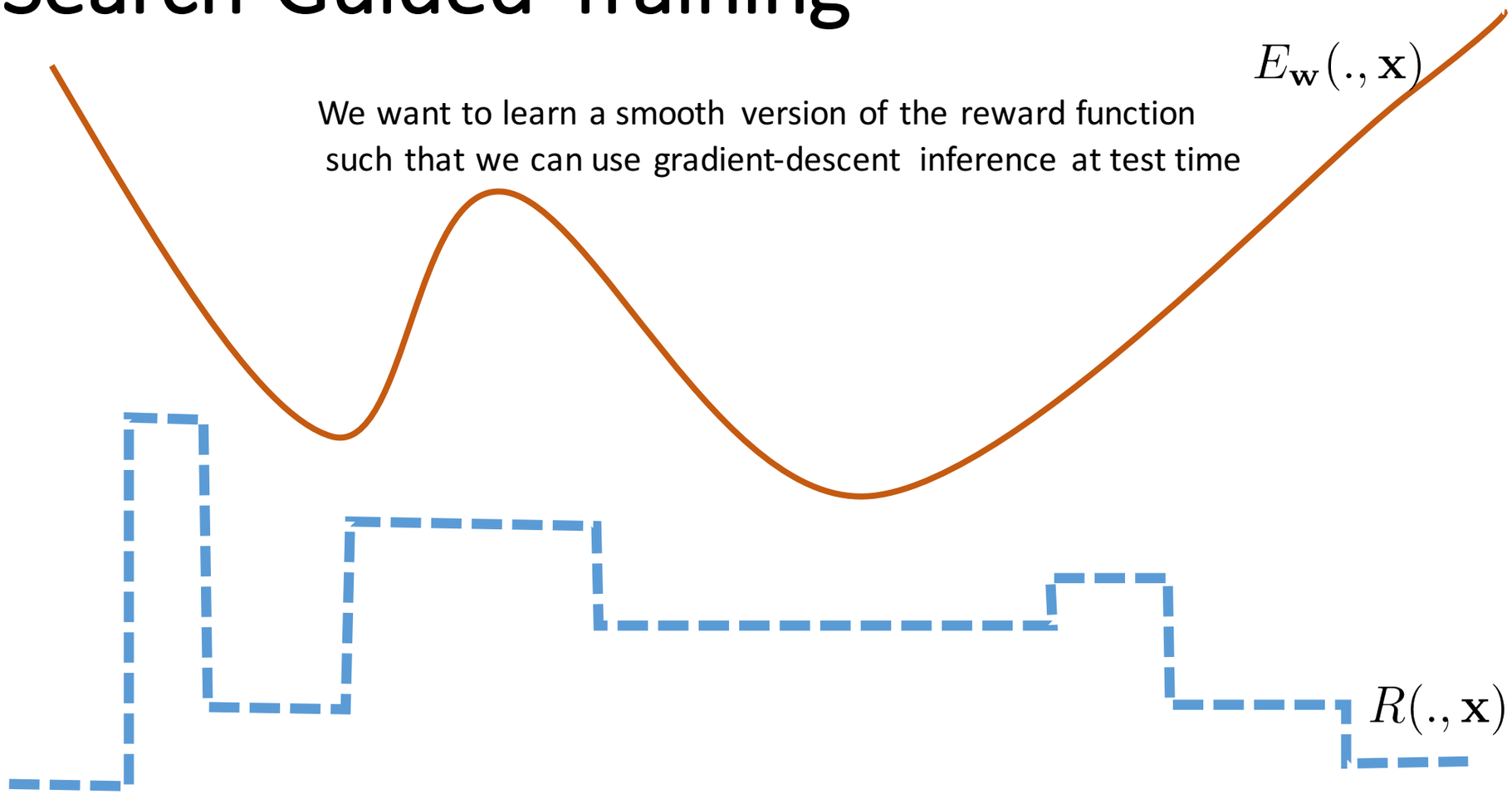
- Data annotation is expensive, especially for structured outputs.
- Domain knowledge as the source of supervision.
  - It can be written as reward functions $R(\mathbf{x}, \mathbf{y})$
- $R(\mathbf{x}, \mathbf{y})$ evaluates a pair of input and output configuration into a scalar value
- For a given x, we are looking for the best y that maximize $R(\mathbf{x}, \mathbf{y})$

Search-Guided Training



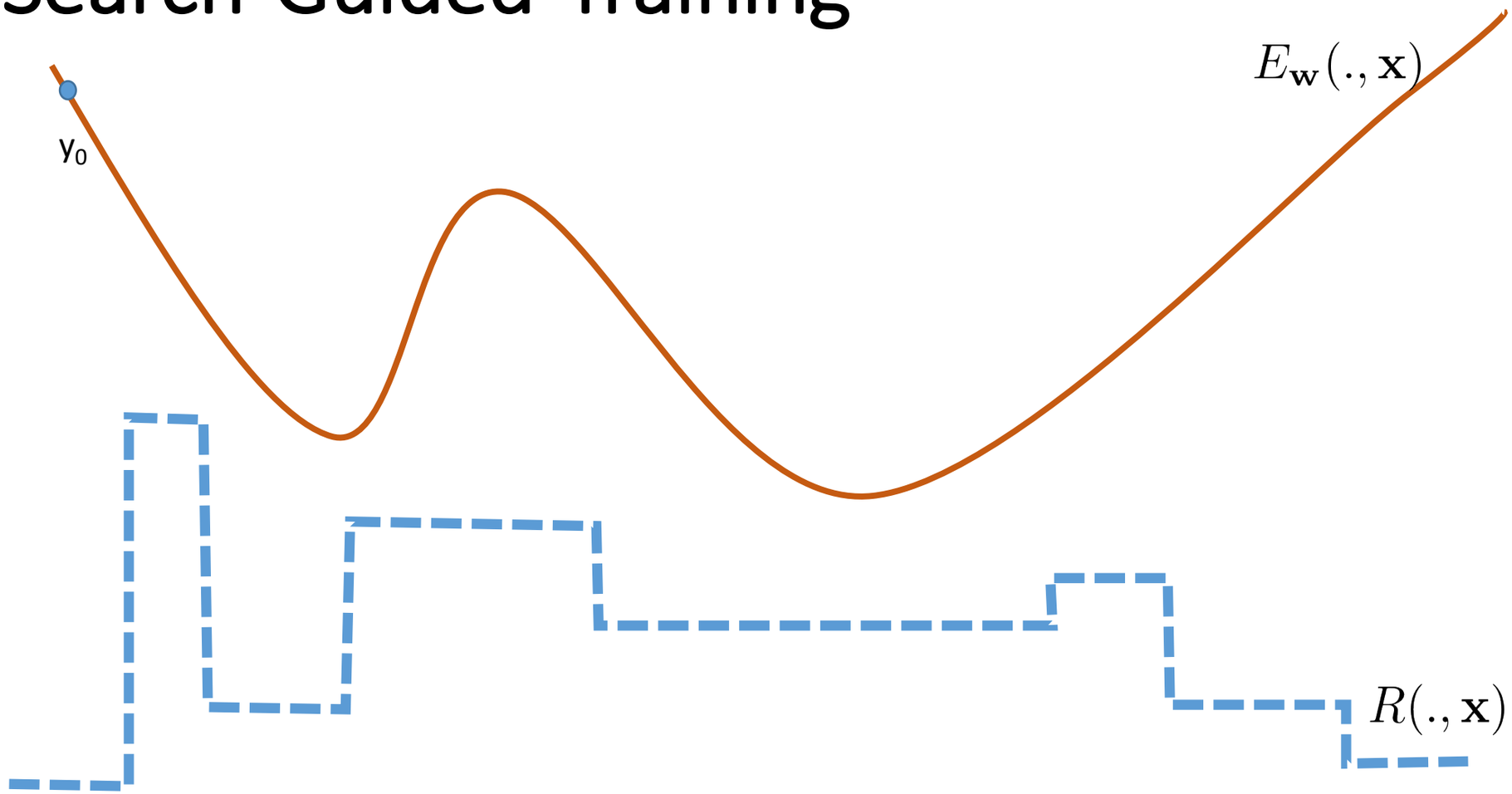
We have a reward function that provides indirect supervision

Search-Guided Training



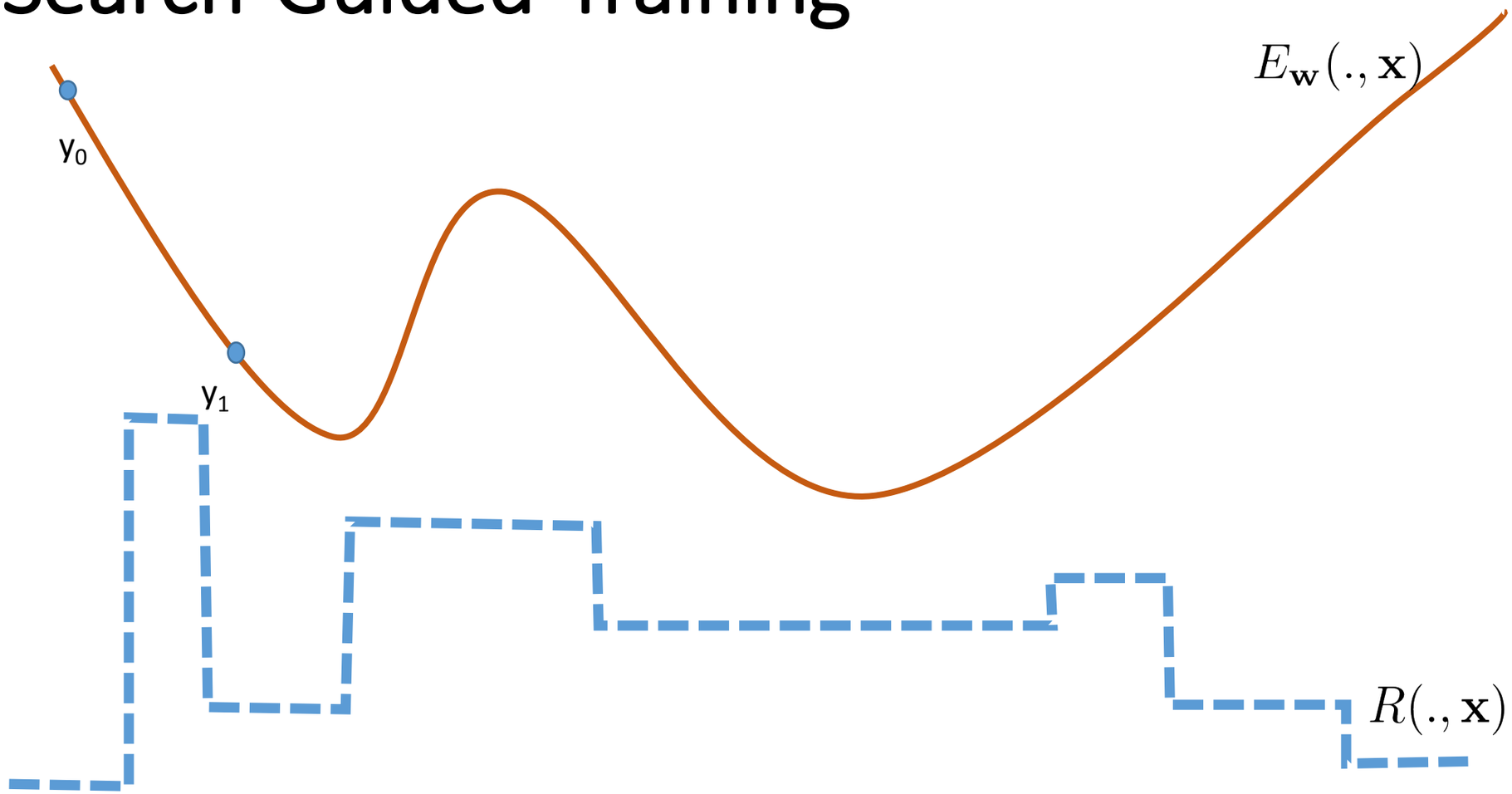
We have a reward function that provides indirect supervision

Search-Guided Training



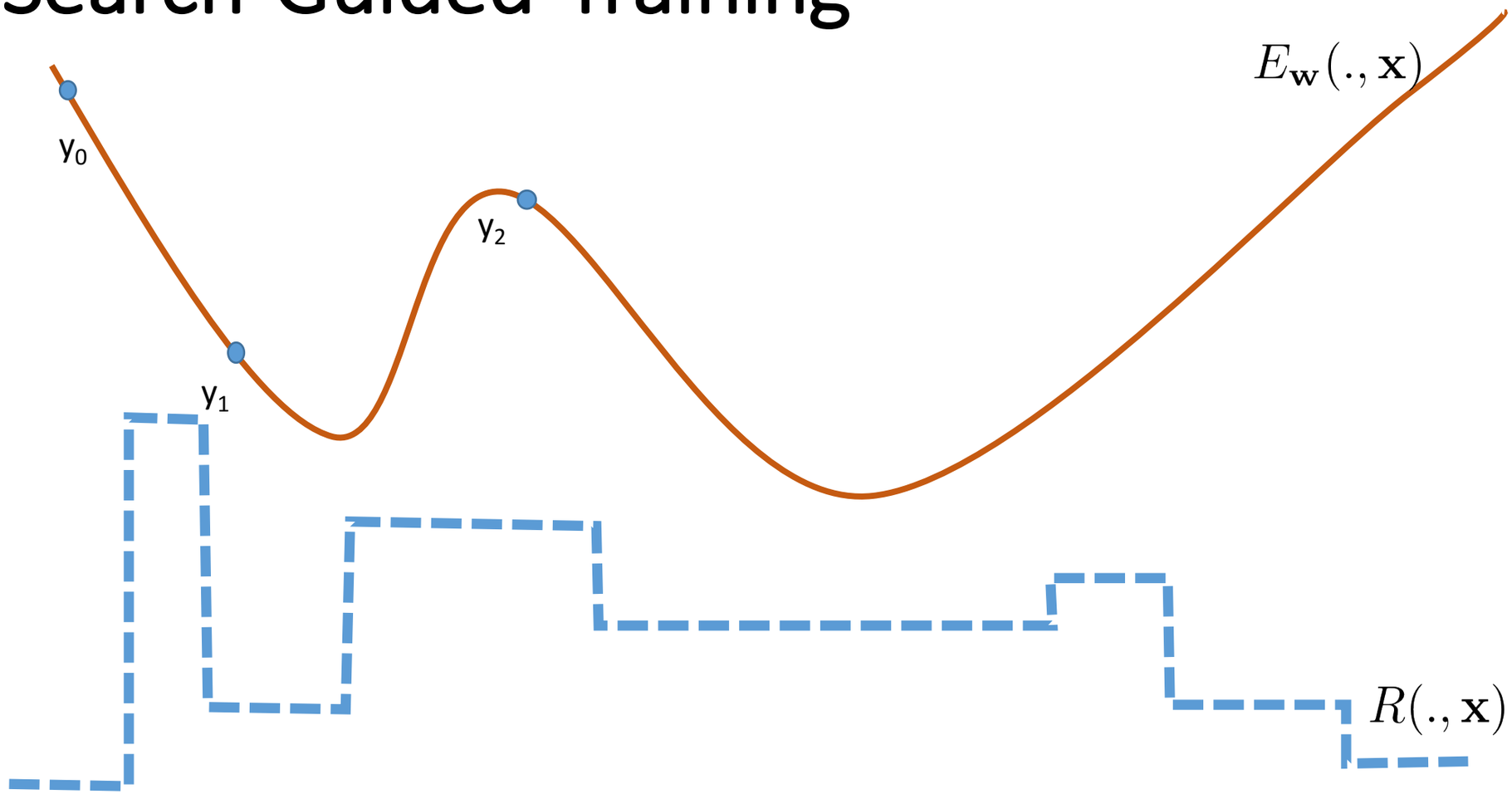
We sample a point from energy function using noisy gradient-descent inference

Search-Guided Training



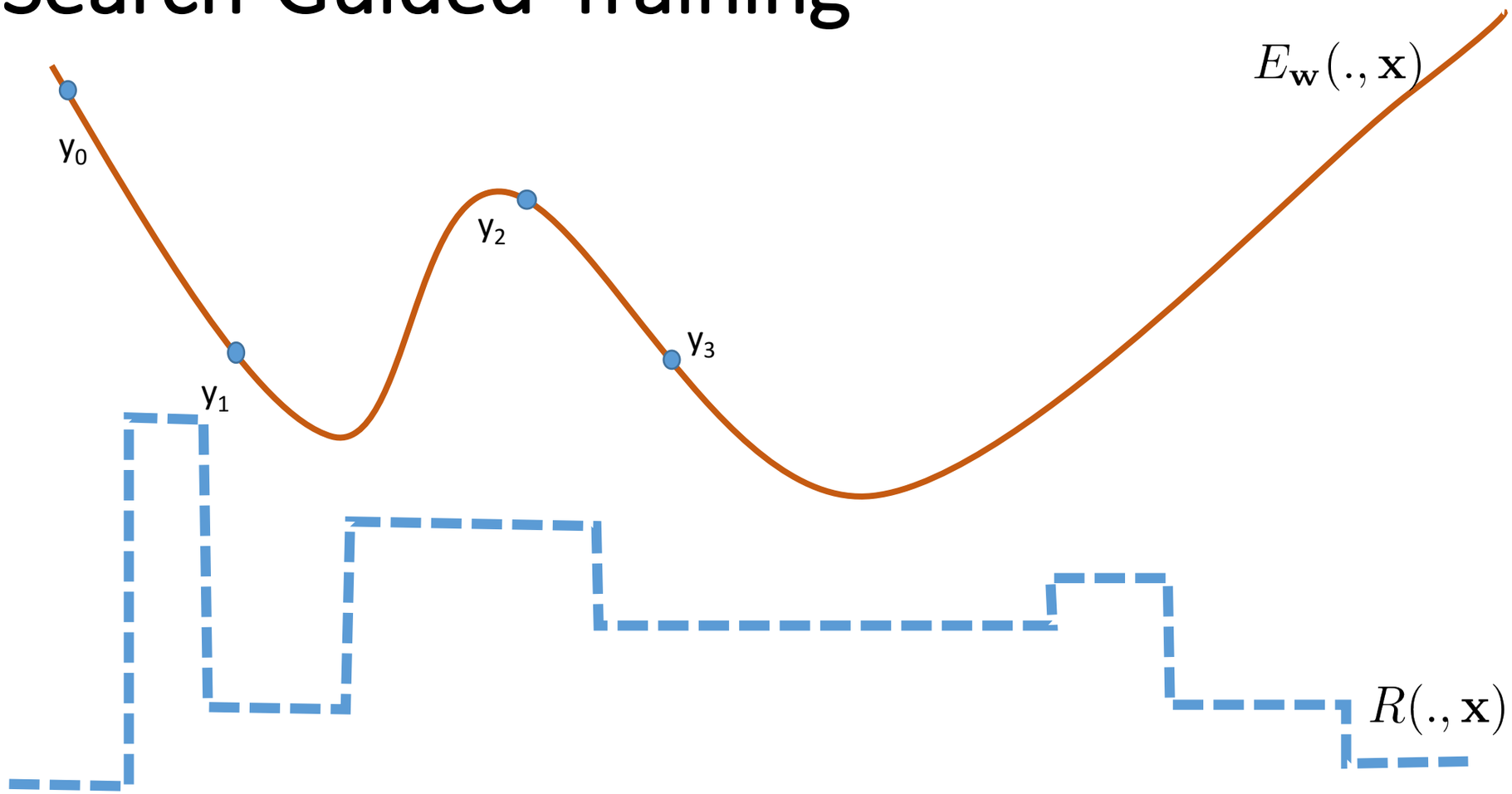
We sample a point from energy function using noisy gradient-descent inference

Search-Guided Training



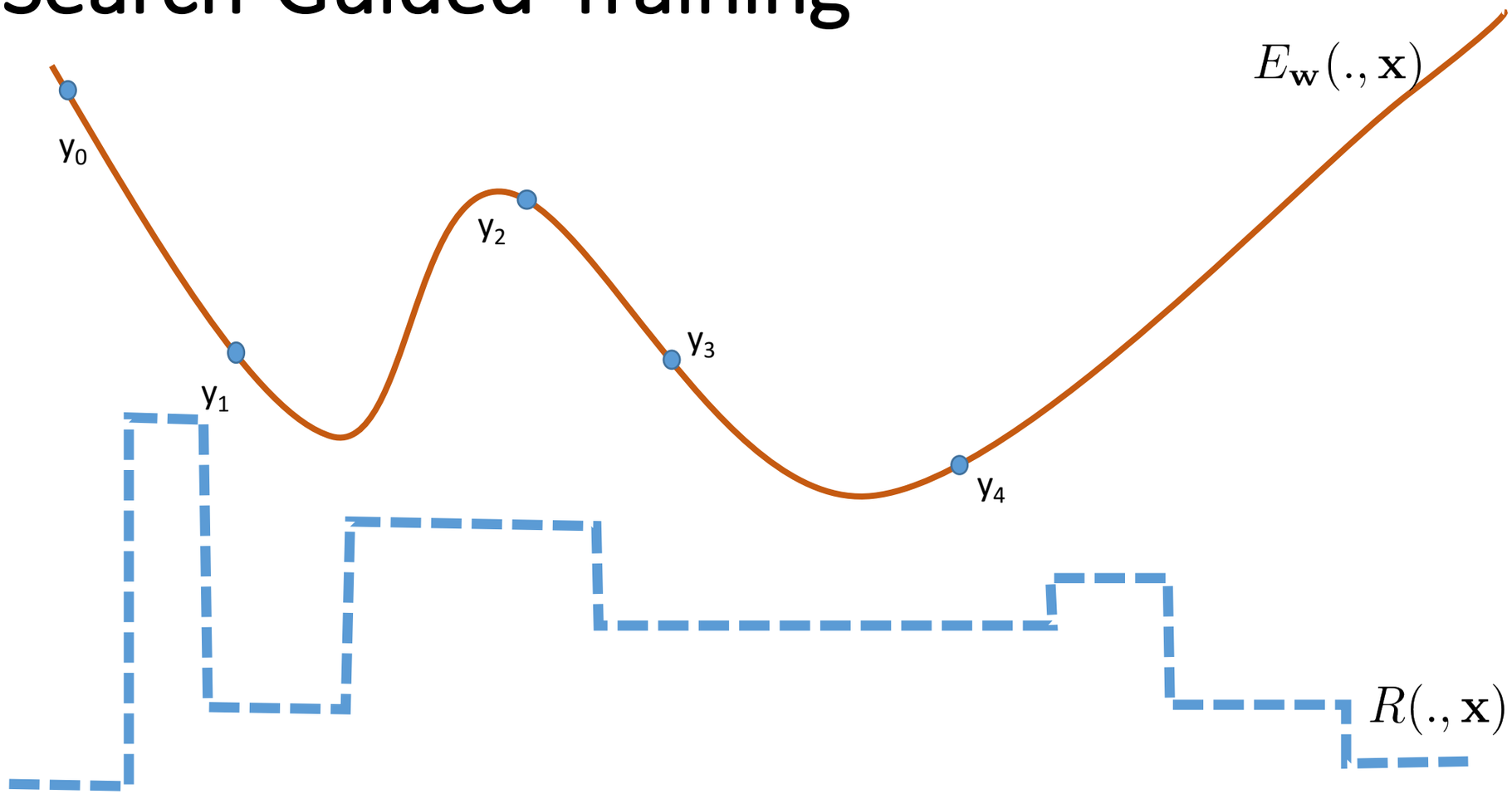
We sample a point from energy function using noisy gradient-descent inference

Search-Guided Training



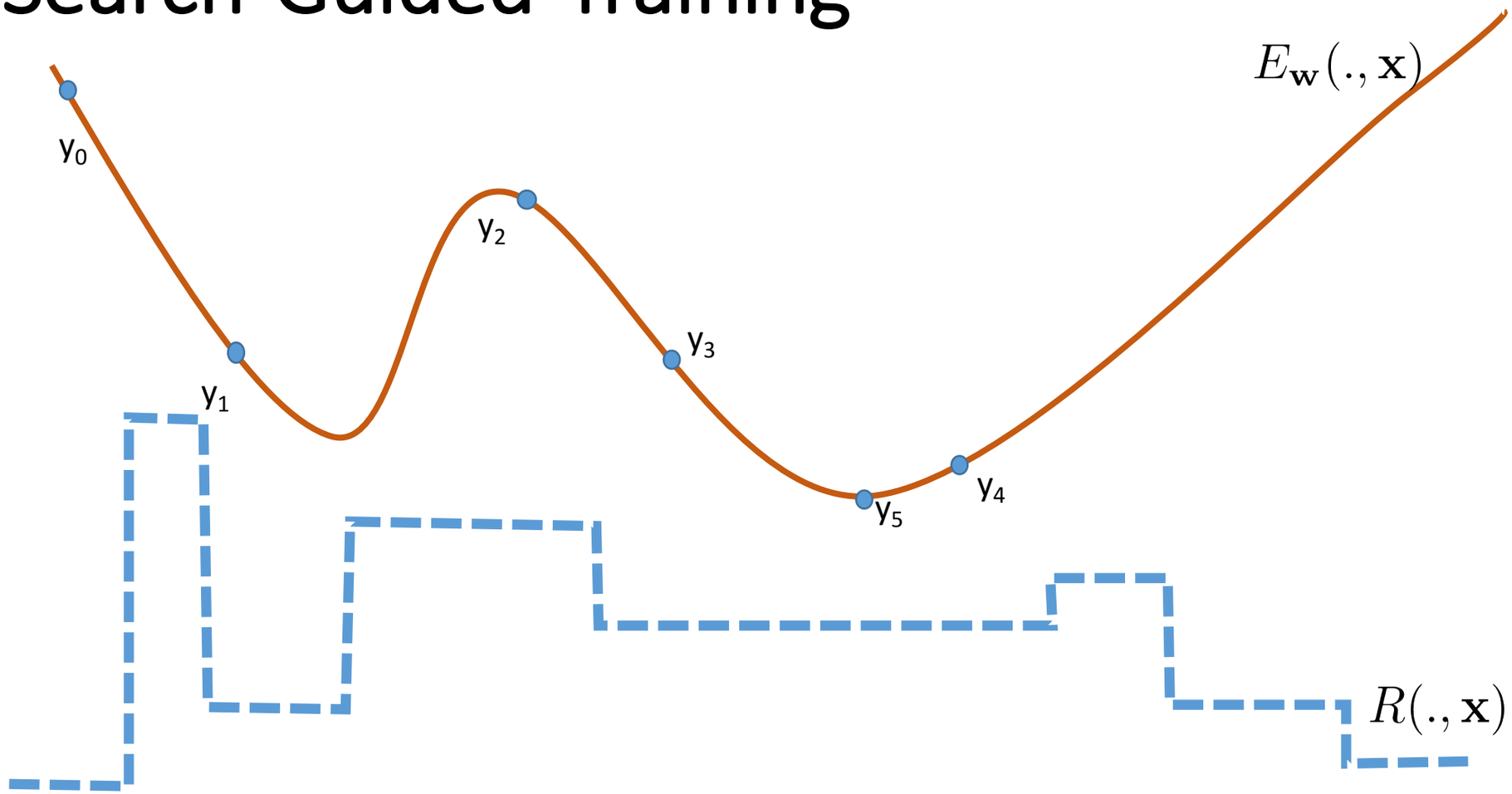
We sample a point from energy function using noisy gradient-descent inference

Search-Guided Training



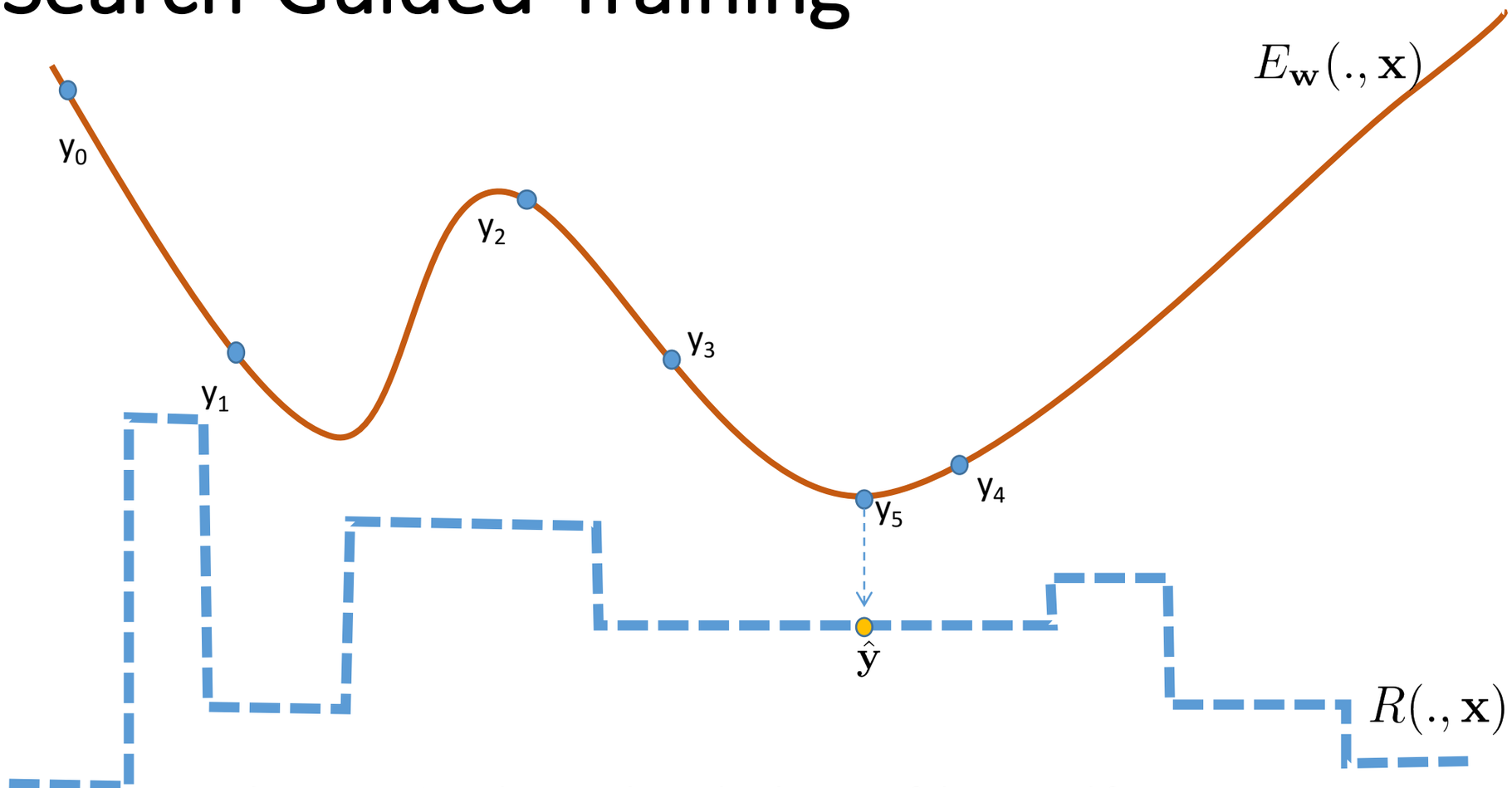
We sample a point from energy function using noisy gradient-descent inference

Search-Guided Training



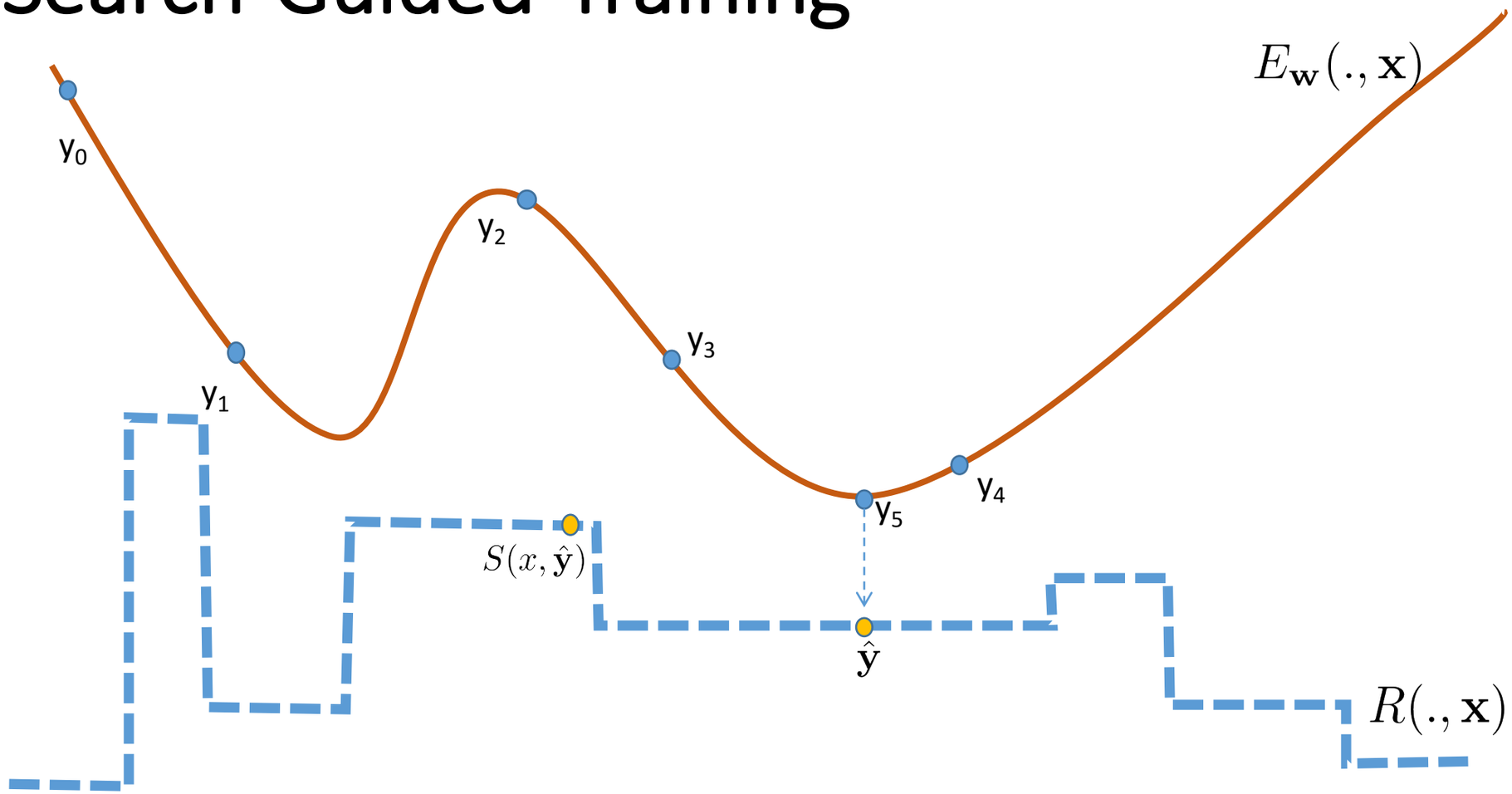
We sample a point from energy function using noisy gradient-descent inference

Search-Guided Training



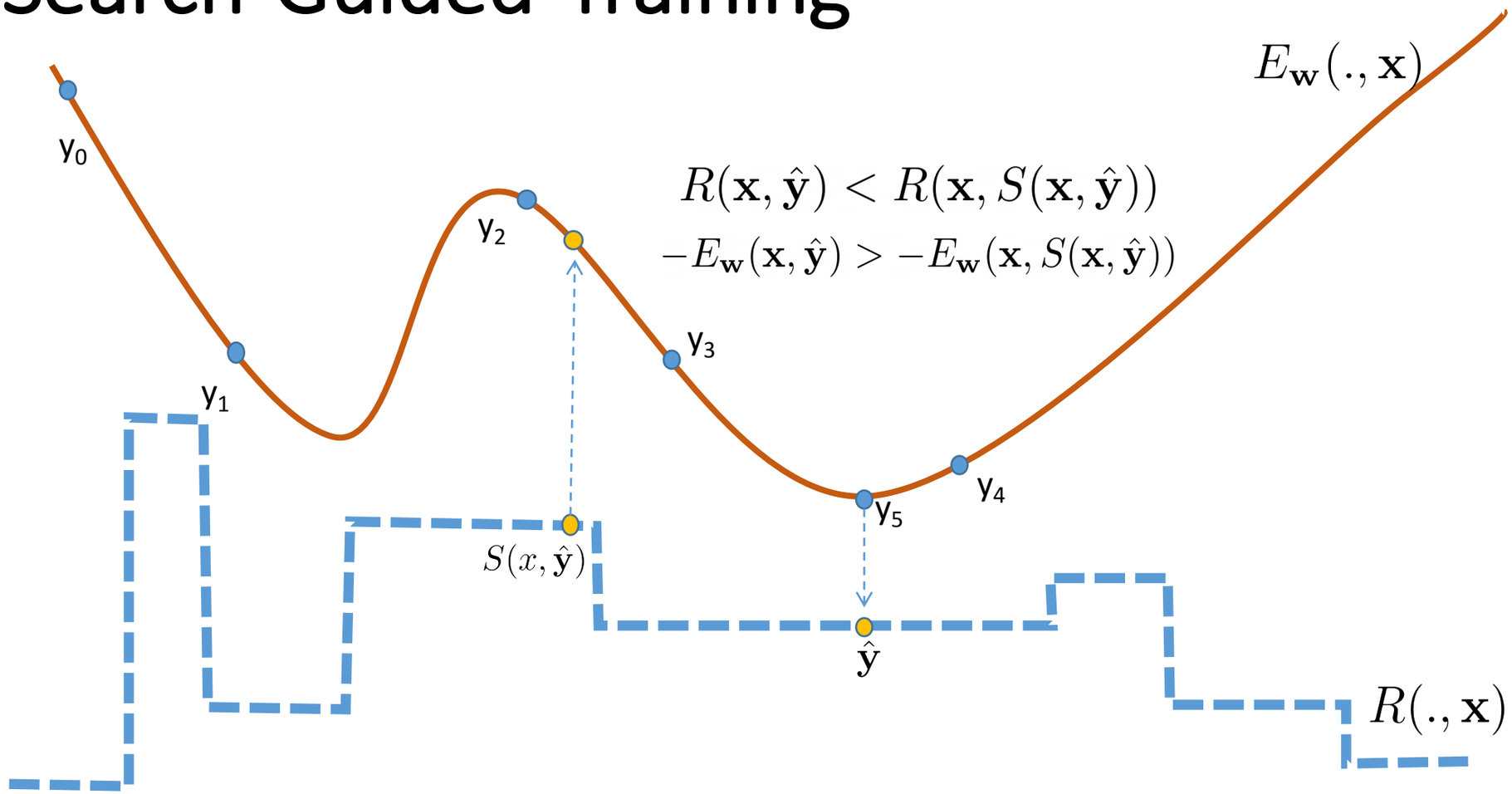
Then we project the sample to the domain of the reward function
(the sample is a point in the simplex,
but the domain of the reward function is often discrete, i.e., the vertices of the simplex)

Search-Guided Training



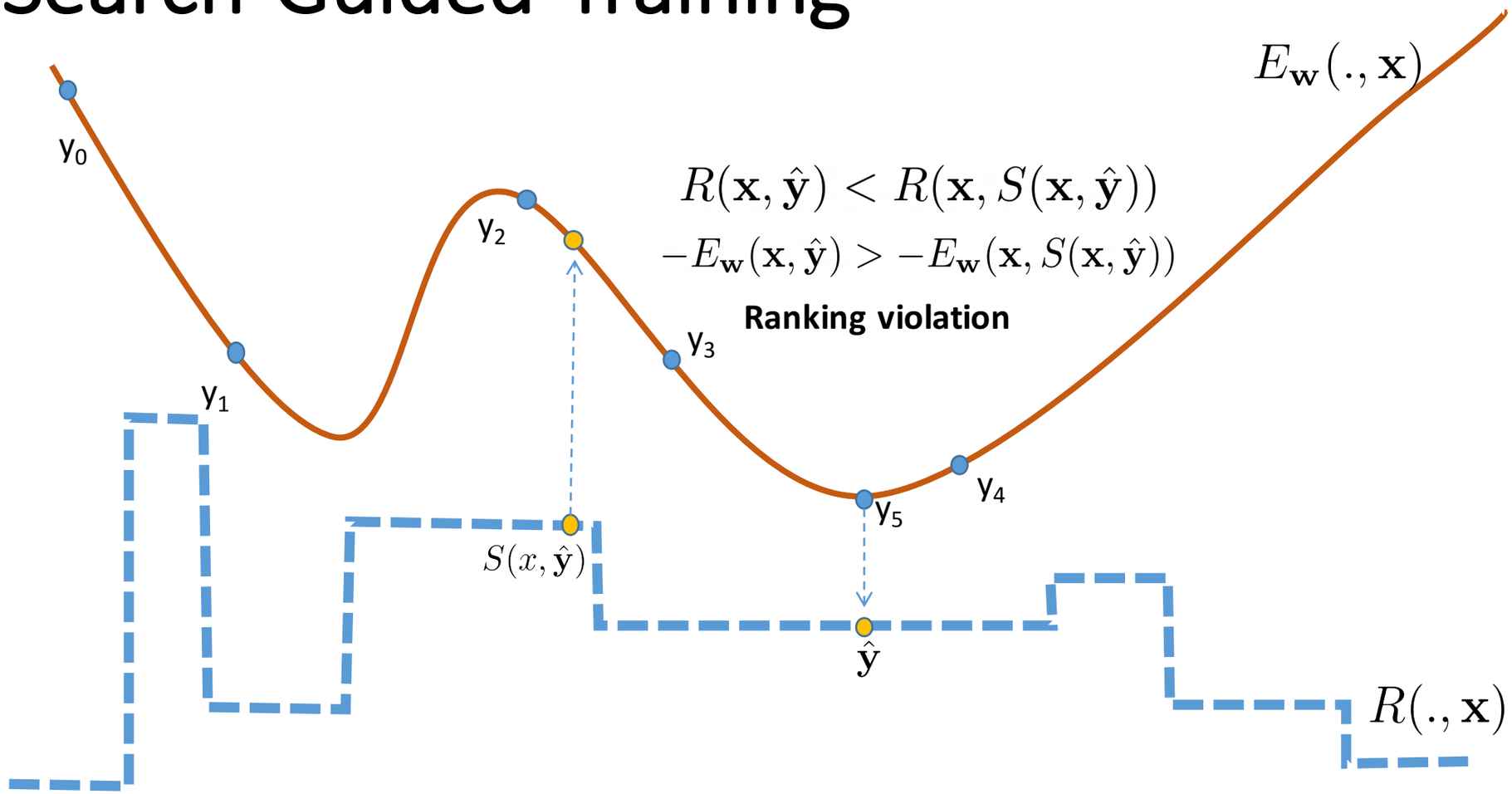
Then the search procedure uses the sample as input and returns an output structure by searching the reward function

Search-Guided Training



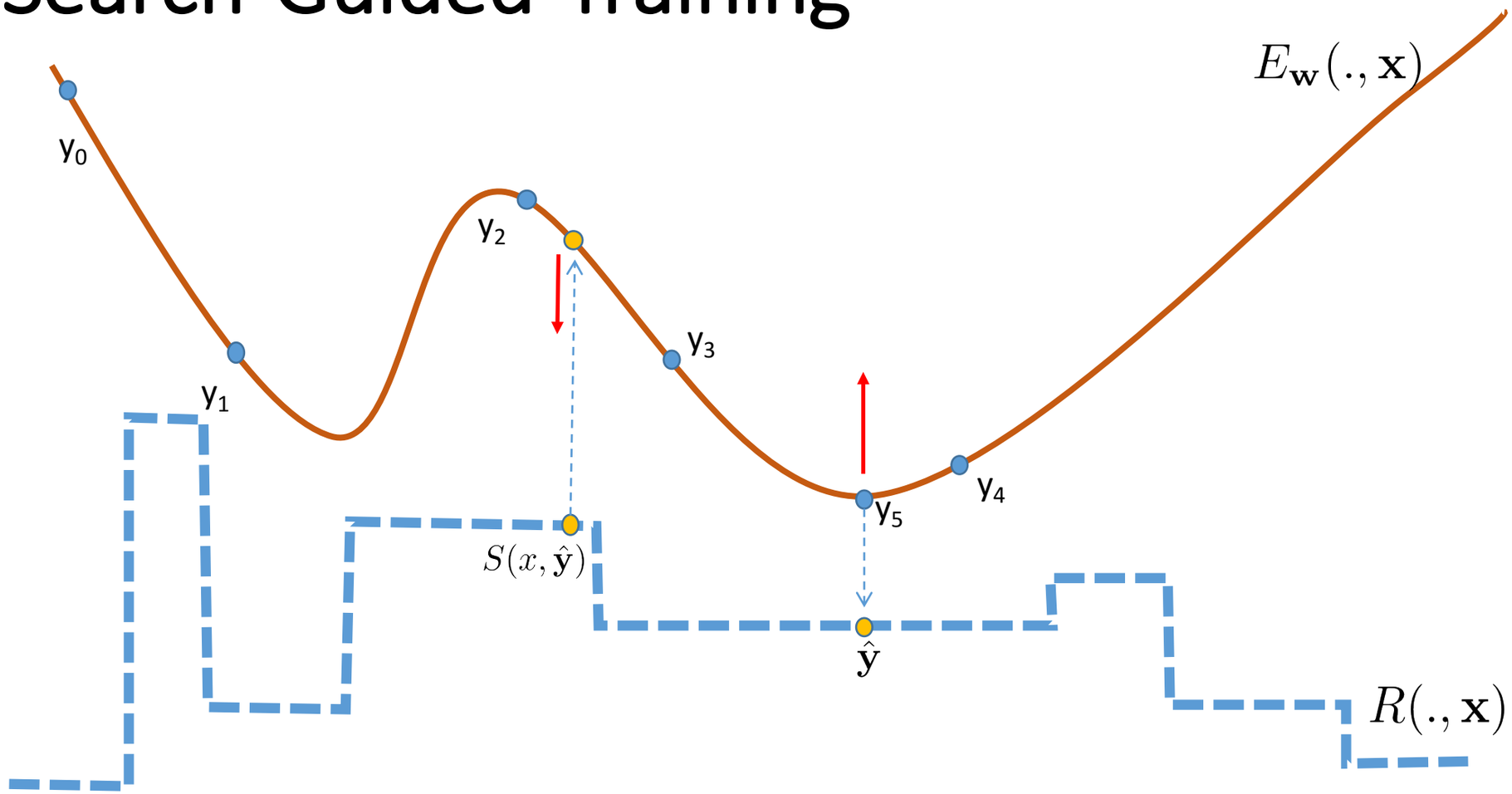
We expect that the two points have the same ranking on the reward function and negative of the energy function

Search-Guided Training



We expect that the two points have the same ranking on the reward function and negative of the energy function

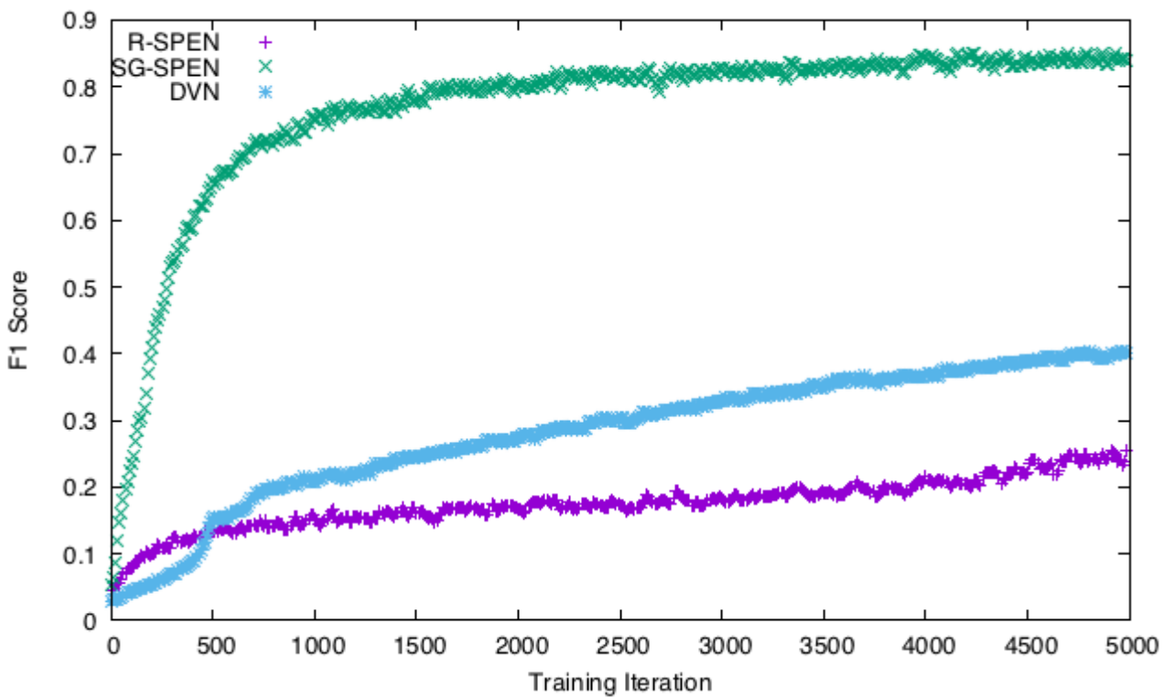
Search-Guided Training

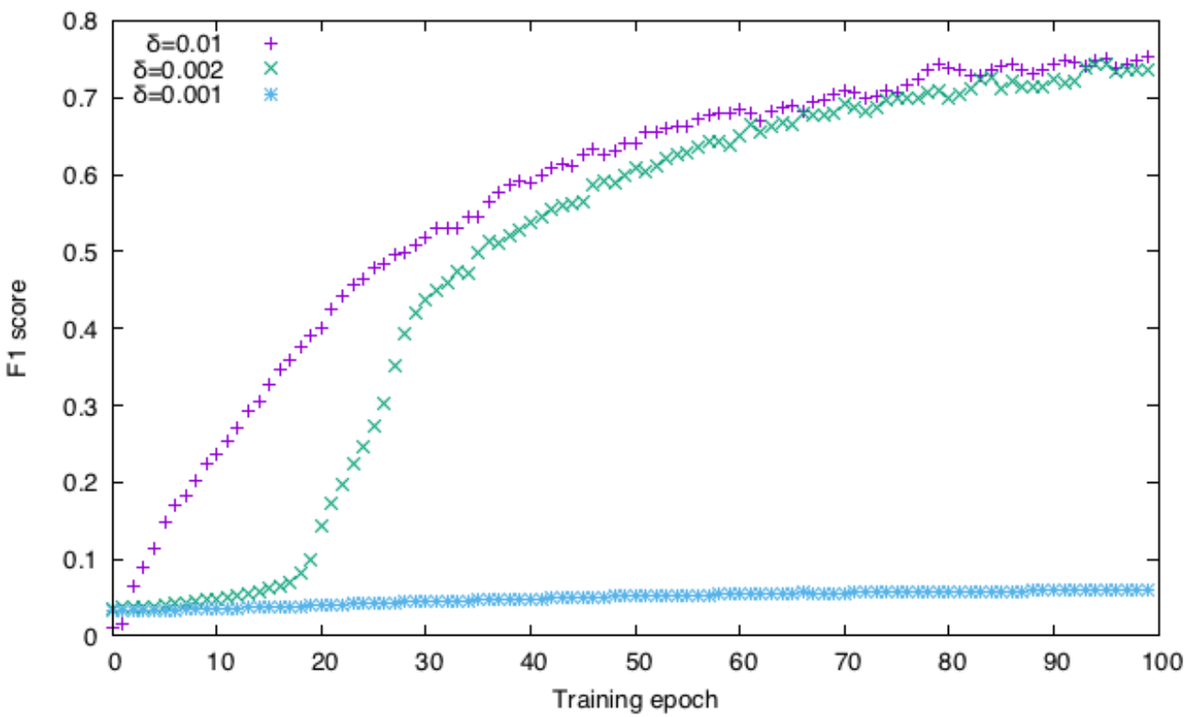


When we find a pair of points that violates the ranking constraints, we update the energy function towards reducing the violation

Task-Loss as Reward Function for Multi-Label Classification

- The simplest form of indirect supervision is to use task-loss as reward function: $R(x, y) = F_1(x, y, y*)$





Domain Knowledge as Reward Function for Citation Field Extraction

x = Citation Token Sequence

```
Warren , D . H . D .
( 1976 ) .
Generating Conditional Plans and Programs .
In Proceedings of the Summer Conference
on AI and Simulation of Behavior
, Edinburgh .
```

y = Seq. of Labels $\in |14|$

```
author author author author author author author author
date date date date
title title title title title title
booktitle booktitle booktitle booktitle booktitle booktitle
booktitle booktitle booktitle booktitle booktitle booktitle
booktitle location location
```

```
score <- Contains the score of each example
first_seen <- Contains the index of
        the first appearance of each tag
j <- Index of the current token
i <- Index of the current example

# Parantheses have the same tag of what comes inside
if j > 0 and last_token == '(' or current_token == ')':
  if tags[j] != tags[j-1]:
    score[i] -= 1

# Period takes that tag of its predecessor
if j > 0 and current_token == '.':
  if j > 0 and tags[j] != tags[j-1]:
    score[i] -= 1

# Only one of the booktitle, journal,
#    or techinical report can appear
if first_seen[booktitle_tag] >= 0 :
  if first_seen[journal_tag] >= 0
      or first_seen[technical_report_tag] >= 0:
    score[i] -= 1

if first_seen[journal_tag] >= 0:
  if first_seen[booktitle_tag] >= 0
      or first_seen[technical_report_tag] >= 0:
    score[i] -= 1

if first_seen[techincal_report_tag] >= 0:
  if first_seen[booktitle_tag] >= 0
      or first_seen[journal_tag] >= 0:
    score[i] -= 1
```

Domain Knowledge as Reward Function for Citation Field Extraction

x = Citation Token Sequence

```
Warren , D . H . D .
( 1976 ) .
Generating Conditional Plans and Programs .
In Proceedings of the Summer Conference
on AI and Simulation of Behavior
, Edinburgh .
```

y = Seq. of Labels ∈ |14|

```
author author author author author author author author
date date date date
title title title title title title
booktitle booktitle booktitle booktitle booktitle booktitle
booktitle booktitle booktitle booktitle booktitle booktitle
booktitle location location
```

```
score <- Contains the score of each example
first_seen <- Contains the index of
        the first appearance of each tag
j <- Index of the current token
i <- Index of the current example

# Parantheses have the same tag of what comes inside
if j > 0 and last_token == '(' or current_token == ')':
  if tags[j] != tags[j-1]:
    score[i] -= 1

# Period takes that tag of its predecessor
if j > 0 and current_token == '.':
  if j > 0 and tags[j] != tags[j-1]:
    score[i] -= 1

# Only one of the booktitle, journal,
#    or techinical report can appear
if first_seen[booktitle_tag] >= 0 :
  if first_seen[journal_tag] >= 0
      or first_seen[technical_report_tag] >= 0:
    score[i] -= 1

if first_seen[journal_tag] >= 0:
  if first_seen[booktitle_tag] >= 0
      or first_seen[technical_report_tag] >= 0:
    score[i] -= 1

if first_seen[techincal_report_tag] >= 0:
  if first_seen[booktitle_tag] >= 0
      or first_seen[journal_tag] >= 0:
    score[i] -= 1
```

Domain Knowledge as Reward Function for Citation Field Extraction

x = Citation Token Sequence

```
Warren , D . H . D .
( 1976 ) .
Generating Conditional Plans and Programs .
In Proceedings of the Summer Conference
on AI and Simulation of Behavior
, Edinburgh .
```

y = Seq. of Labels ∈ |14|

```
author author author author author author author author
date date date date
title title title title title title
booktitle booktitle booktitle booktitle booktitle booktitle
booktitle booktitle booktitle booktitle booktitle booktitle
booktitle location location
```

```
score <- Contains the score of each example
first_seen <- Contains the index of
        the first appearance of each tag
j <- Index of the current token
i <- Index of the current example

# Parantheses have the same tag of what comes inside
if j > 0 and last_token == '(' or current_token == ')':
  if tags[j] != tags[j-1]:
    score[i] -= 1

# Period takes that tag of its predecessor
if j > 0 and current_token == '.':
  if j > 0 and tags[j] != tags[j-1]:
    score[i] -= 1

# Only one of the booktitle, journal,
#    or techinical report can appear
if first_seen[booktitle_tag] >= 0 :
  if first_seen[journal_tag] >= 0
      or first_seen[technical_report_tag] >= 0:
    score[i] -= 1

if first_seen[journal_tag] >= 0:
  if first_seen[booktitle_tag] >= 0
      or first_seen[technical_report_tag] >= 0:
    score[i] -= 1

if first_seen[techincal_report_tag] >= 0:
  if first_seen[booktitle_tag] >= 0
      or first_seen[journal_tag] >= 0:
    score[i] -= 1
```

Domain Knowledge as Reward Function for Citation Field Extraction

x = Citation Token Sequence

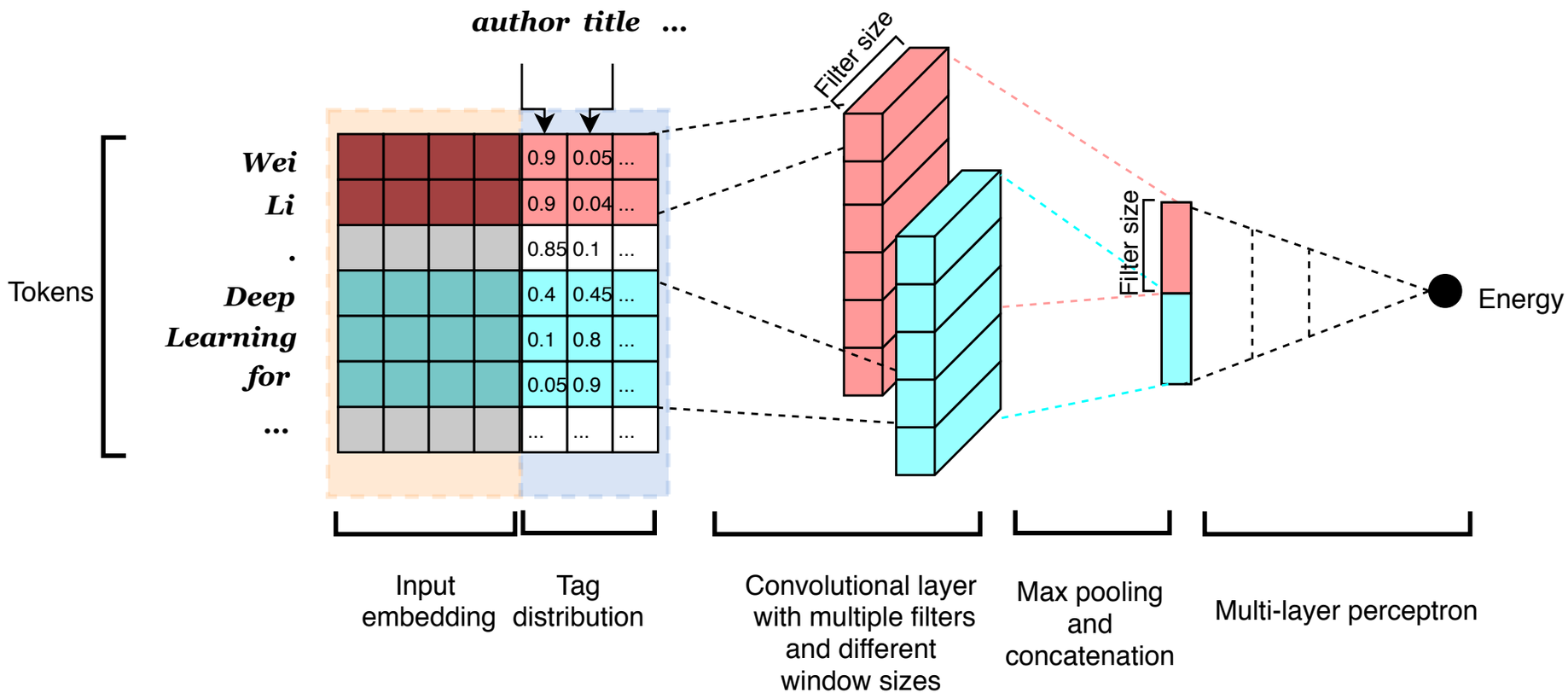
```
Warren , D . H . D .
( 1976 ) .
Generating Conditional Plans and Programs .
In Proceedings of the Summer Conference
on AI and Simulation of Behavior
, Edinburgh .
```

y = Seq. of Labels ∈ |14|

```
author author author author author author author author
date date date date
title title title title title title
booktitle booktitle booktitle booktitle booktitle booktitle
booktitle booktitle booktitle booktitle booktitle booktitle
booktitle location location
```

```
score <- Contains the score of each example
first_seen <- Contains the index of
        the first appearance of each tag
j <- Index of the current token
i <- Index of the current example

# Parantheses have the same tag of what comes inside
if j > 0 and last_token == '(' or current_token == ')':
  if tags[j] != tags[j-1]:
    score[i] -= 1

# Period takes that tag of its predecessor
if j > 0 and current_token == '.':
  if j > 0 and tags[j] != tags[j-1]:
    score[i] -= 1

# Only one of the booktitle, journal,
#    or techinical report can appear
if first_seen[booktitle_tag] >= 0 :
  if first_seen[journal_tag] >= 0
      or first_seen[technical_report_tag] >= 0:
    score[i] -= 1

if first_seen[journal_tag] >= 0:
  if first_seen[booktitle_tag] >= 0
      or first_seen[technical_report_tag] >= 0:
    score[i] -= 1

if first_seen[techincal_report_tag] >= 0:
  if first_seen[booktitle_tag] >= 0
      or first_seen[journal_tag] >= 0:
    score[i] -= 1
```

Energy Model



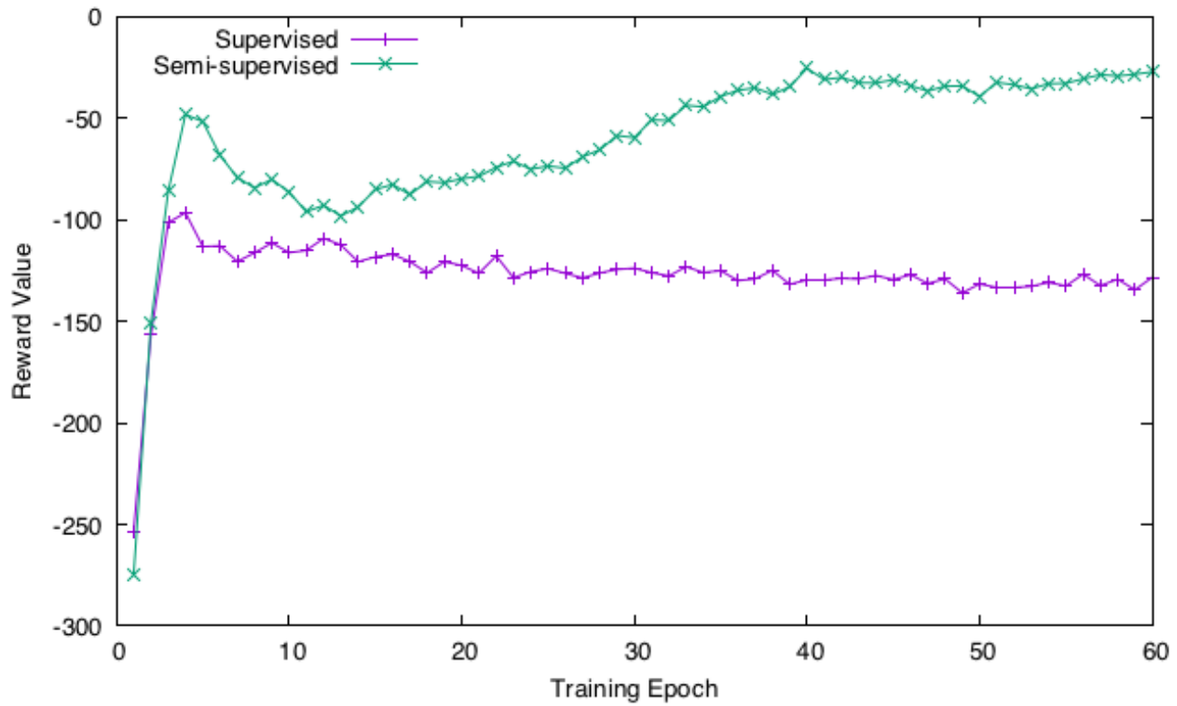
Performance on Citation Field Extraction

Method	Accuracy	Inference time (sec.)
GE	37.3%	-
Iterative Beam Search		
K=1	30.5%	159
K=2	35.7%	850
K=5	39.3%	2,892
K=10	39.0%	6,654
PG		
EMA baseline	54.5%	< 1
Parametric baseline	47.9%	< 1
MMRN	39.5%	< 1
DVN	29.6%	< 1
R-SPEN	48.3%	< 1
SG-SPEN	**57.1**%	< 1

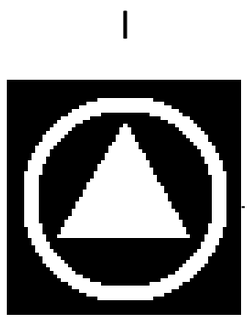
Semi-Supervised Setting

- Alternatively use the output of search and ground-truth label for training.

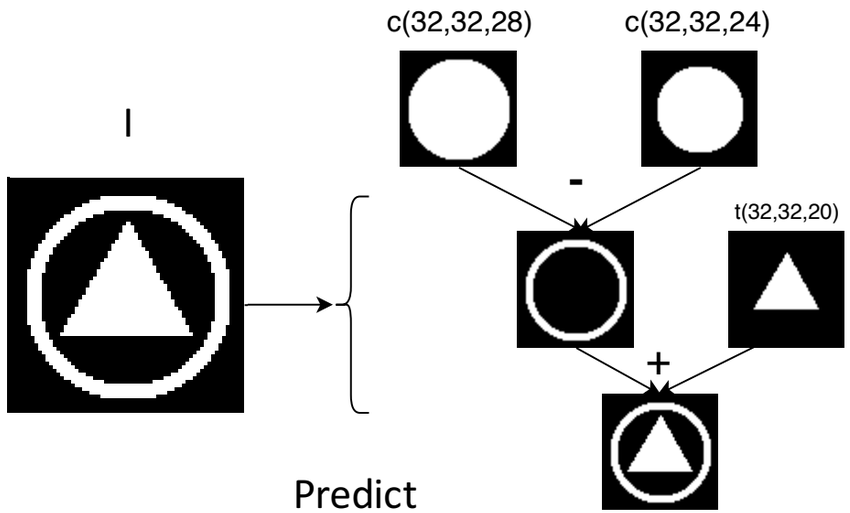
No.	GE	PG	DVN	R-SPEN	SG-SPEN	SG-SPEN-sup	DVN-sup
5	54.7	55.6	50.5	55.0	**65.5**	53.0	57.4
10	57.9	67.7	60.6	65.5	**71.7**	62.4	61.9
50	68.0	76.5	67.7	81.5	**82.9**	81.6	81.4



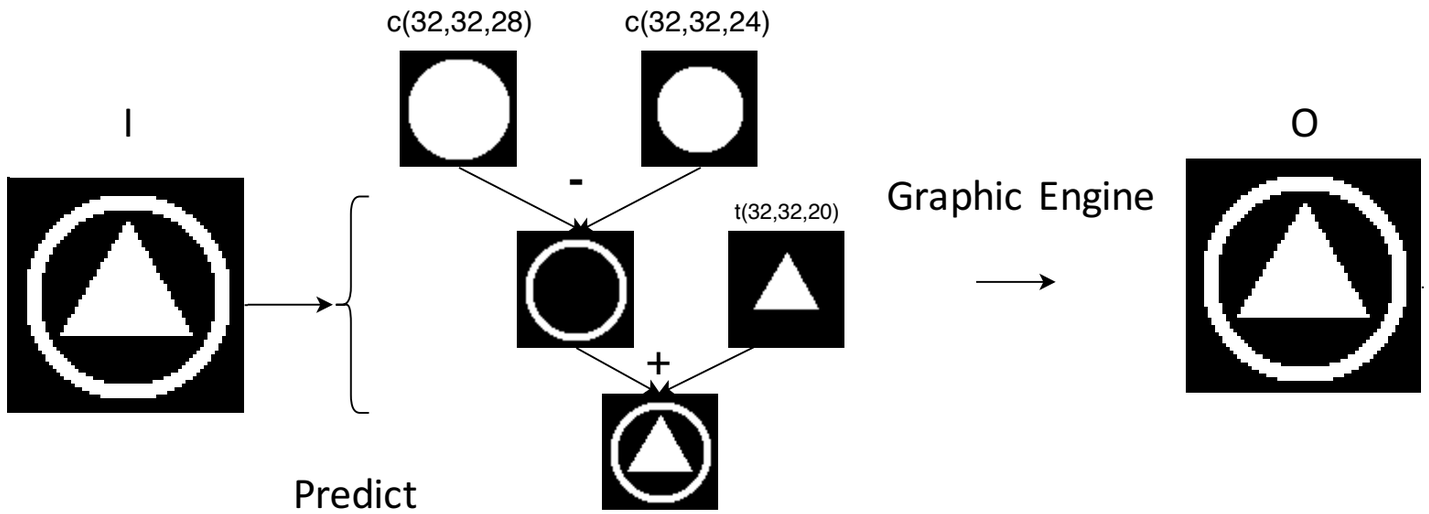
Shape Parser



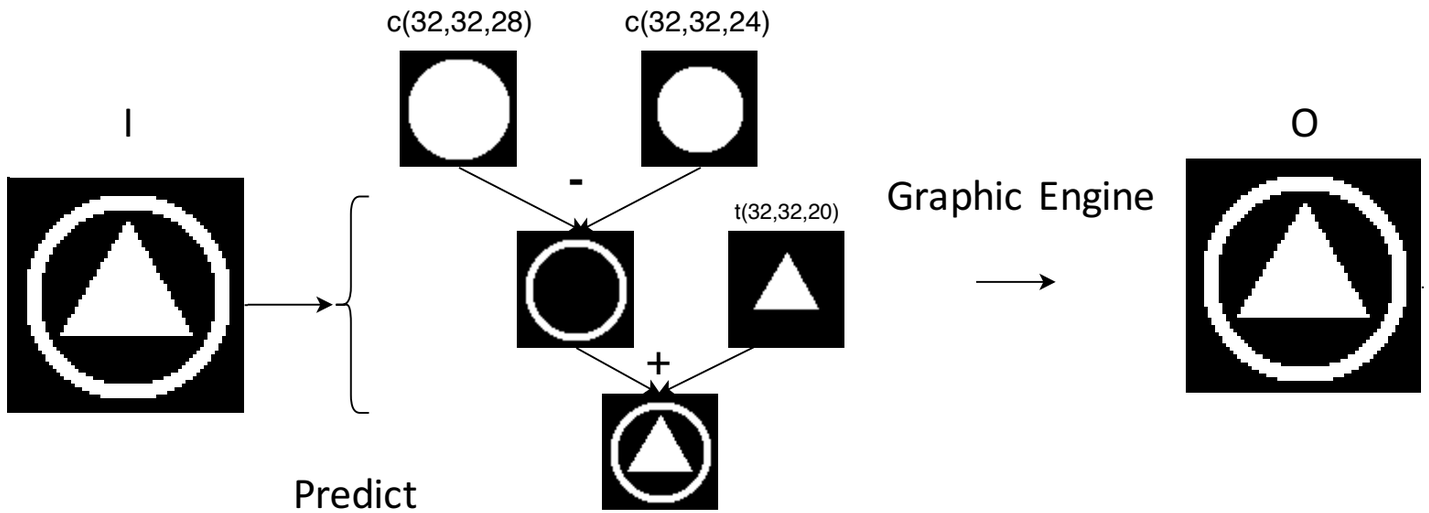
Shape Parser



Shape Parser

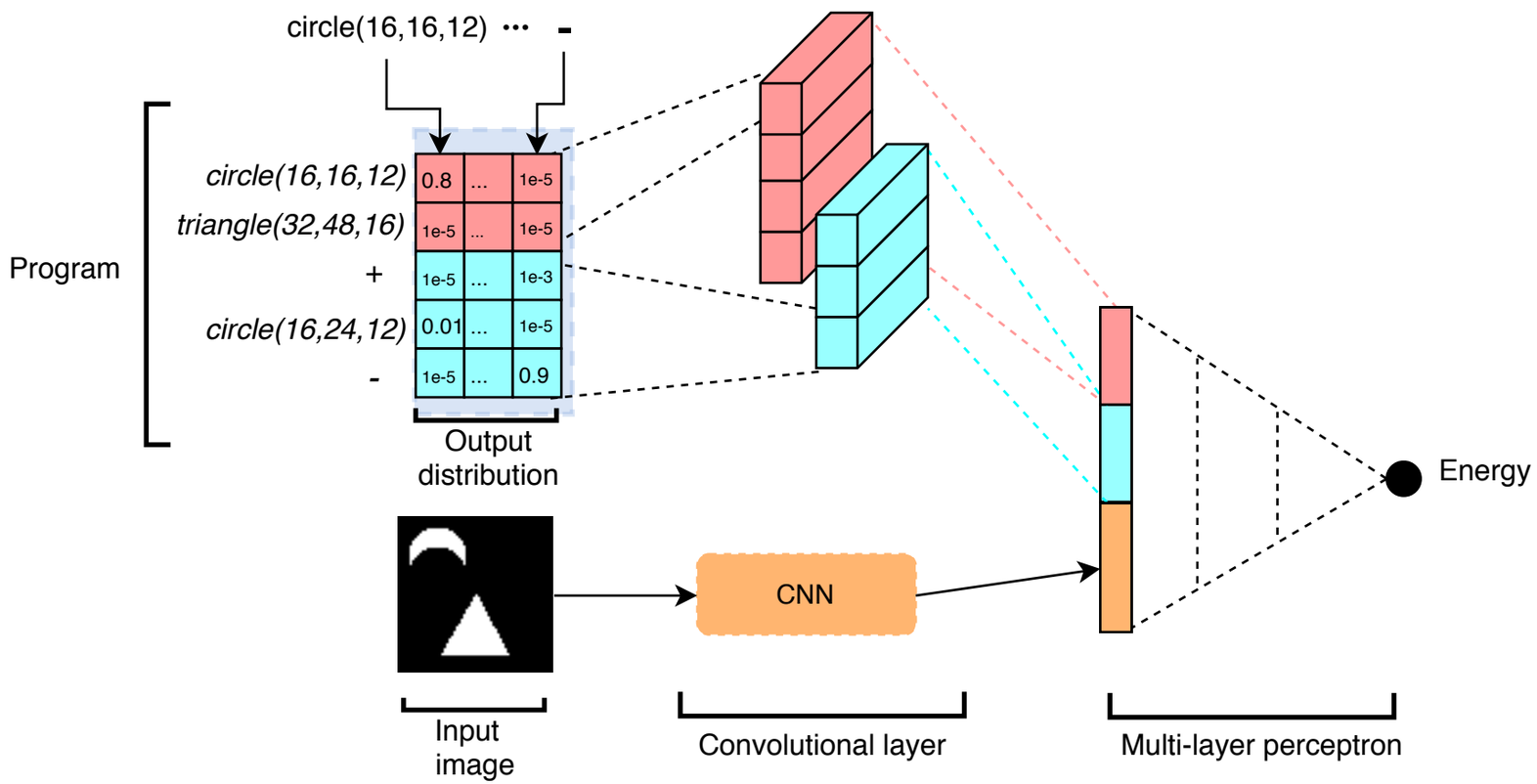


Shape Parser

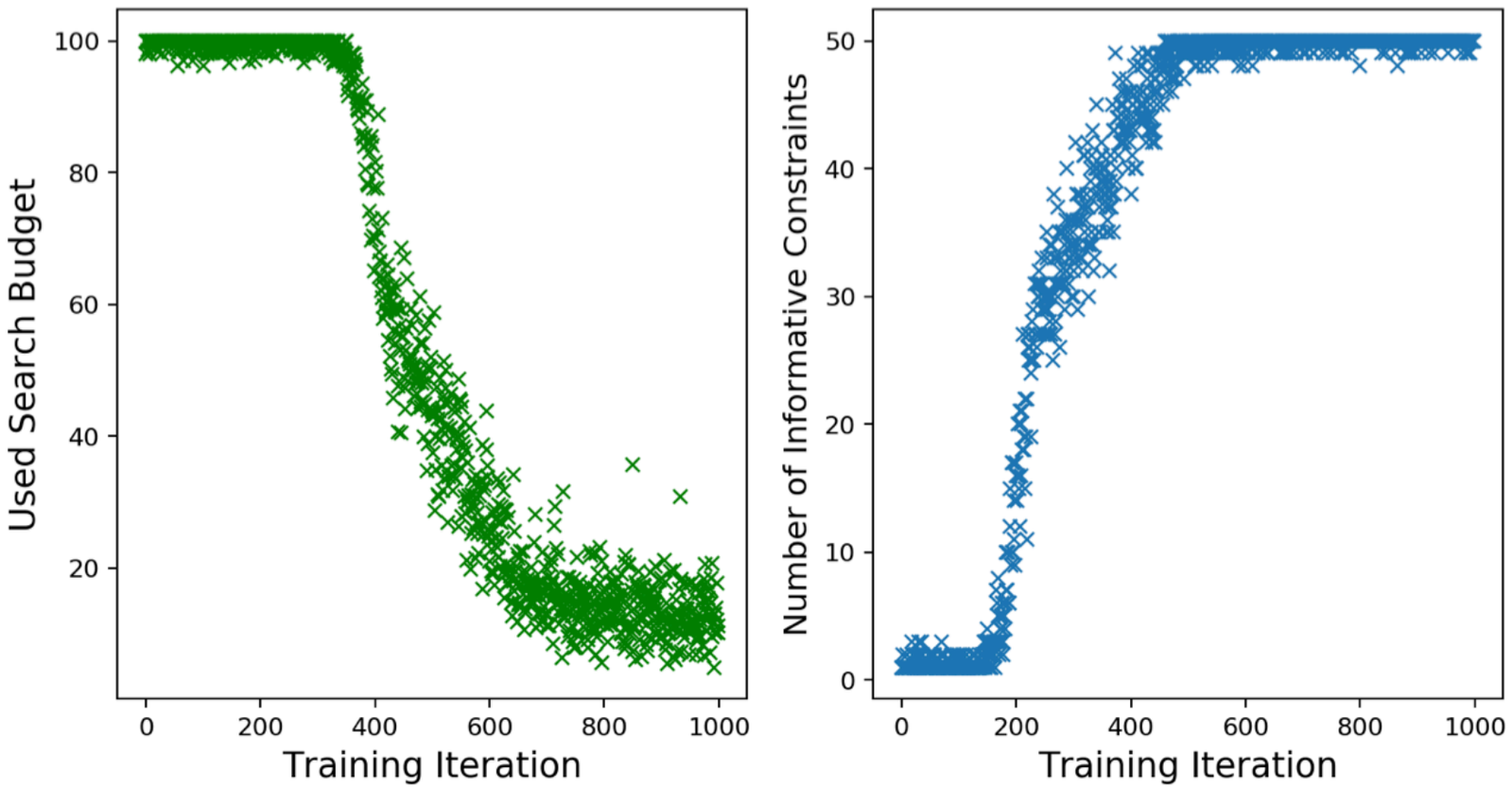


$$R(I,O) = \frac{I \cap O}{I \cup O}$$

Shape Parser Energy Model



Search Budget vs. Constraints



Performance on Shape Parser

Method	IOU	Inference time (sec.)
Iterative Beam Search		
K=5	24.6%	3,882
K=10	30.0%	15,537
K=20	43.1%	38,977
Neural shape parser	32.4%	< 1
SG-SPEN	**56.3**%	< 1

Conclusion and Future Directions

- If a reward function exists to evaluate every structured output into a scalar value
  - We can use unlabled data for training structured prediction energy networks
- Domain knowledge or non-differentiable pipelines can be used to define the reward functions.
- The main ingredient for learning from the reward function is the search operator.
- Here we only use simple search operators, but more complex search functions derived from domain knowledge can be used for complicated problems.