Grounding Linguistic Analysis in Control Applications

S.R.K. Branavan

February 8, 2012
Right click "My Computer" on the desktop, and click the Manage menu option.

Click Services after expanding "Services and applications"

Cities built on or near water sources can irrigate to increase their crop yields.
Our Goal

Leverage language-control connection to:

• *Learn language analysis from control feedback*

• *Improve control performance using textual information*

*Text interpretation can enable novel automation*
Semantic Interpretation: Traditional Approach

Map text into an abstract representation

List flights to Boston on Friday night.
\[ \lambda x. \text{flight}(x) \land \text{to}(x, \text{bos}) \land \text{day}(x, \text{fri}) \land \text{during}(x, \text{night}) \]

\[ [\text{The man}]_{Arg0} \text{ opened } [\text{the door}]_{Arg1} \text{ him} [\text{today}]_{Arg3} [\text{today}]_{ArgM-TMP}. \]

(Typical papers on semantics)
Semantic Interpretation: Our Approach

*Map text to control actions*

**Text**

Build your city on grassland with a river running through it if possible.

**Control actions**

- \( \text{MOVE	extunderscore TO} (7,3) \)
- \( \text{BUILD	extunderscore CITY} () \)
- ...

Enables language learning from control feedback
Build your city on grassland with a river running through it if possible.

Text:

Control actions

MOVE_TO (7, 3)
BUILD_CITY ()

Higher game score

(7, 3) is grassland with a river

Control actions

MOVE_TO (5, 1)
BUILD_CITY ()

Lower game score

(5, 1) is a desert

Learn Language via Reinforcement Learning
Challenges

• Situational Relevance

*Relevance of textual information depends on control state*

“Cities built on or near water sources can irrigate to increase their crop yields, and cities near mineral resources can mine for raw materials.”

• Abstraction

*Text can describe abstract concepts in the control application*

“Build your city on grassland.”
“Water is required for irrigation.”

• Incompleteness

*Text does not provide complete information about the control application*

“Build your city on grassland with a river running through it if possible.”
(what if there are no rivers nearby?)
Contributions

- Learn language analysis from control feedback
- Improve control performance using text information

Instruction Interpretation
Complex Game-play
High-level Planning
General Setup

Input:
• Text documents
• Interactive access to control application

Prior knowledge:
• Text provides information useful for the control application

Goal:
• Learn to interpret the text and learn effective control
Outline

1. Step-by-step imperative instructions
   - *Learning from control feedback*

2. High-level strategy descriptions
   - *Situational relevance*
   - *Incompleteness*
   - *Learning from control feedback*

3. General descriptions of world dynamics
   - *Abstractions*
   - *Situational relevance*
   - *Incompleteness*
   - *Learning from control feedback*
1. Step-by-step imperative instructions
   - Learning from control feedback

2. High-level strategy descriptions
   - Situational relevance
   - Incompleteness
   - Learning from control feedback

3. General descriptions of world dynamics
   - Abstractions
   - Situational relevance
   - Incompleteness
   - Learning from control feedback
**Interpreting Instructions to Actions**

**Input**

**Instructions:**
step-by-step descriptions of commands

**Target environment:**
where commands need to be executed

**Output**

**Command sequence**
executable in the environment

1. Click `Start`, point to `Search`, and then click `For Files or Folders`.
2. In the `Search for` box, type "msdownld.tmp"
3. In the `Look in` list, click `My Computer`, and then click `Search Now`.
4. ...

```
LEFT_CLICK(Start)
LEFT_CLICK(Search)
...
TYPE_INTO(Search for: , "msdownld.tmp")
...
```
Instruction Interpretation: Challenges

Segment text to chunks that describe individual commands

Select run after clicking start, then in the open box, type "dcomcnfg".

Learn translation of words to environment commands

"select run" \(ightarrow\) LEFT_CLICK \(\rightarrow\) run

Reorder environment commands

1. clicking start, then
2. Select run after
3. in the open box, type "dcomcnfg".
Instruction Interpretation: Representation

Markov Decision Process - select text segment, translate & execute:

1. click Run, and press OK after typing secpol.msc in the open box.
   left-click [Run...]

2. click Run, and press OK after typing secpol.msc in the open box.
   left-click [Run...] type-into [open] "secpol.msc"

3. click Run, and press OK after typing secpol.msc in the open box.
   left-click [Run...] type-into [open] "secpol.msc" left-click [OK]
Instruction Interpretation: Representation

\[ \text{State} \quad s = \text{Observed Text} + \text{Observed Environment} \]

\[ \text{Action} \quad a = \text{Word Selection} + \text{Environment Command} \]

State 1

Observed text and environment

Select run after clicking start. In the open box type “dcomcnfg”.

Action 1

words: clicking start
command: \( \text{LEFT\_CLICK( start )} \)

State 2

Observed text and environment

Select run after clicking start. In the open box type “dcomcnfg”.

Policy function

\[ p(a \mid s) \]
Model Parameterization

Represent each action with a feature vector:

\[ \phi(s, a) \in \mathbb{R}^n \] - real valued feature function on state \( s \) and action \( a \)

Define policy function as a log-linear distribution:

\[
p(a \mid s; \theta) = \frac{e^{\theta \cdot \phi(s, a)}}{\sum_{a'} e^{\theta \cdot \phi(s, a')}}
\]

\( \theta \) - parameters of model
Reward Signal

Ideal:

Test for task completion

Alternative Indication of Error:

Text specifies objects not visible on the screen

Approximation:

If a sentence matches no GUI labels, a preceding action is wrong
Learning Using Reward Signal: Challenges

1. Reward can be delayed

How can reward be propagated to individual actions

2. Number of candidate action sequences is very large

How can this space be effectively searched?

Use Reinforcement Learning
Learning Algorithm

Goal: Find $\theta$ that maximizes the expected reward

Method: Policy gradient algorithm (stochastic gradient ascent on $\theta$)

for each document:

sample candidate action sequence:

- observe world state $s_t$
- select action $a_t \sim p(a|s_t; \theta)$
- execute action in world
- receive reward $r$
- update parameters $\theta$ based on reward

Document text:

1. Click Start, point to Search, and then click File.
2. In the Search Results dialog box, on the Tools menu, click Show hidden files and folders, and then click extensions for known file types check box.
3. In the Folder Options dialog box, on the View tab, click Show hidden files and folders, and then click extensions for known file types check box.
4. In the Search for files or folders named box, type Search.
5. In the Look in list, click My Computer, and then click Search.
6. In the Search Results pane, right-click My Computer, and then click Search the shortcut menu.
7. Click Yes.

Environment:

Open
Explore
Search...
Open All Users
Explore All Users
Start
Windows Configuration Application

Windows 2000 help documents from support.microsoft.com

<table>
<thead>
<tr>
<th>Total # of documents</th>
<th>128</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train / development / test</td>
<td>70 / 18 / 40</td>
</tr>
<tr>
<td>Total # of words</td>
<td>5562</td>
</tr>
<tr>
<td>Vocabulary size</td>
<td>610</td>
</tr>
<tr>
<td>Avg. words per sentence</td>
<td>9.93</td>
</tr>
<tr>
<td>Avg. sentences per document</td>
<td>4.38</td>
</tr>
<tr>
<td>Avg. actions per document</td>
<td>10.37</td>
</tr>
</tbody>
</table>
Results: Command Accuracy

- **Majority heuristic**: 29%
- **Control feedback**: 79%
- **Manual supervision**: 79%

% commands correctly mapped
Outline

1. Step-by-step imperative instructions
   - Learning from control feedback

2. High-level strategy descriptions
   - Situational relevance
   - Incompleteness
   - Learning from control feedback

3. General descriptions of world dynamics
   - Abstractions
   - Situational relevance
   - Incompleteness
   - Learning from control feedback
Solving Hard Decision Tasks

Civilization II Player’s Guide

You start with two settler units. Although settlers are capable of performing a variety of useful tasks, your first task is to move the settlers to a site that is suitable for the construction of your first city. Use settlers to build the city on grassland with a river running through it.

---

How to Load Pallets

1. Place a pallet near the boxes you are loading.
2. Carefully stack boxes in a uniform fashion onto the pallet.
3. Stretch wrap the boxes on the pallet to ensure they do not shift when you move the pallet.

---

Warren Buffett’s Priceless Investment Advice

By John Reese | More Articles
February 12, 2010 | Comments (0)

“It’s far better to buy a wonderful company at a fair price than a fair price.

If you can grasp this simple advice from Warren Buffett, you should there are other investment strategies out there, but Buffett’s approach demonstrably successful over more than 50 years. Why try anything else?

Two words for the efficient market hypothesis: Warren Buffett
Solving Hard Decision Tasks

**Objective:** Maximize a utility function

**Challenge:** Finding optimal solution hard
  - *Large decision space*
  - *Expensive simulations*

**Traditional solution:** Manually encoded domain-knowledge

**Our goal:** Automatically extract required domain knowledge from text
Adversarial Planning Problem

Civilization II: Complex multiplayer strategy game (branching factor $\approx 10^{20}$)

Traditional Approach: Monte-Carlo Search Framework

- Learn action selection policy from simulations
- Very successful in complex games like Go and Poker
Leveraging Textual Advice: Challenges

1. Find sentences relevant to given game state.

**Game state**

**Strategy document**

You start with two settler units. Although settlers are capable of performing a variety of useful tasks, your first task is to move the settlers to a site that is suitable for the construction of your first city. Use settlers to build the city on grassland with a river running through it if possible. You can also use settlers to irrigate land near your city. In order to survive and grow ...
Leveraging Textual Advice: Challenges

1. Find sentences relevant to given game state.

**Game state**

You start with two settler units. Although settlers are capable of performing a variety of useful tasks, your first task is to move the settlers to a site that is suitable for the construction of your first city. Use settlers to build the city on grassland with a river running through it if possible.

You can also use settlers to **irrigate land near your city.** In order to survive and grow ...

**Strategy document**
You start with two settler units. Although settlers are capable of performing a variety of useful tasks, your first task is to move the settlers to a site that is suitable for the construction of your first city. Use settlers to **build the city on grassland with a river running through it if possible.** You can also use settlers to irrigate land near your city. In order to survive and grow ...
Leveraging Textual Advice: Challenges

2. Label sentences with predicate structure.

Move the settler to a site suitable for building a city, onto grassland with a river if possible.

Label words as **action**, **state** or **background**
Leveraging Textual Advice: Challenges

2. Label sentences with predicate structure.

Move the settler to a site suitable for building a city, onto grassland with a river if possible.

Label words as **action**, **state** or **background**.
2. Label sentences with predicate structure.

Move the settler to a site suitable for building a city, onto grassland with a river if possible.

Label words as action, state or background.
Leveraging Textual Advice: Challenges

3. Guide action selection using relevant text

- **Build the city on plains or grassland with a river running through it if possible.**

- $a_1$ – move_settlers_to (7,3)
- $a_2$ – settlers_build_city ()
- $a_3$ – settlers_irrigate_land ()
Learning from Game Feedback

**Goal:** Learn from game feedback as only source of supervision.

**Key idea:** Better parameter settings will lead to more victories.

You start with two settler units. Although settlers are capable of performing a variety of useful tasks, your first task is to move the settlers to a site that is suitable for the construction of your first city. Use settlers to build the city on plains or grassland with a river running through it if possible. In order to survive and grow ...

---

**Model params:**

1. $\theta_1$

---

**Model params:**

2. $\theta_2$

---

**Game manual**

End result: won

End result: lost
Model Overview

Monte-Carlo Search Framework

- Learn action selection policy from simulations
- Very successful in complex games like Go and Poker.

Our Algorithm

- Learn text interpretation from simulation feedback
- Bias action selection policy using text
Monte-Carlo Search

Select actions via simulations, game and opponent can be stochastic

Actual Game

Simulation

Copy game

Irrigate

Game lost

State 1

???
Monte-Carlo Search

Try many candidate actions from current state & see how well they perform.
Monte-Carlo Search

Try many candidate actions from current state & see how well they perform.
Learn feature weights from simulation outcomes

\[ Q(s, a) \propto \vec{\theta} \cdot \vec{\phi}(s, a) \]

- \( \vec{\phi}(s, a) \) - feature function
- \( \vec{\theta} \) - model parameters

<table>
<thead>
<tr>
<th>Rollout depth</th>
<th>State 1 game state</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>5 1 0 1 1 0 1</td>
</tr>
<tr>
<td>0.4</td>
<td>15 0 1 0 0 1 0</td>
</tr>
<tr>
<td>1.2</td>
<td>37 1 0 1 0 0 0</td>
</tr>
<tr>
<td>3.5</td>
<td>...............................</td>
</tr>
</tbody>
</table>

Game scores

- 0.1
- 0.4
- 1.2
- 3.5
Model Overview

Monte-Carlo Search Framework

• *Learn action selection policy from simulations*

Our Algorithm

• *Bias action selection policy using text*

• *Learn text interpretation from simulation feedback*
Modeling Requirements

- Identify sentence relevant to game state
  - Build cities near rivers or ocean.

- Label sentence with predicate structure
  - Build cities near rivers or ocean.

- Estimate value of candidate actions
  - Irrigate: -10
  - Fortify: -5
  - Build city: 25
Sentence Relevance

Identify sentence relevant to game state and action

State $s$, candidate action $a$, document $d$

$p(y = y_i | s, a, d) \propto e^{\vec{u} \cdot \vec{\phi}(y_i, s, a, d)}$

Sentence $y_i$ is selected as relevant

Log-linear model: $\left\{ \begin{align*} \vec{u} & \quad - \text{weight vector} \\ \vec{\phi}(y_i, s, a, d) & \quad - \text{feature function} \end{align*} \right.$
Predicate Structure

Select word labels based on sentence + dependency info

E.g., “Build cities near rivers or ocean.”

Log-linear model:

\[
p(e_j | j, y, q) \propto e^{\vec{v} \cdot \vec{\psi}(e_j, j, y, q)}
\]

Word index \( j \), sentence \( y \), dependency info \( q \)

Predicate label \( e_j = \{ \text{action, state, background} \} \)

\( \vec{v} \) - weight vector

\( \vec{\psi}(e_j, j, y, q) \) - feature function
Final Q function approximation

**Predict expected value of candidate action**

State $s$, candidate action $a$

$$Q(s, a, d, y_i, \tilde{e}_i) = \tilde{w} \cdot \tilde{f}(s, a, d, y_i, \tilde{e}_i)$$

Document $d$, relevant sentence $y_i$, predicate labeling $\tilde{e}_i$

**Linear model:**

$$\begin{cases} 
\tilde{w} & \text{- weight vector} \\
\tilde{f}(s, a, d, y_i, \tilde{e}_i) & \text{- feature function}
\end{cases}$$
**Model Representation**

Multi-layer neural network: *Each layer represents a different stage of analysis*

**Input:**
- game state
- candidate action
- document text

**Q function approximation**

\[ Q(s, a, d, y_i, e_i) = \hat{w} \cdot \hat{f}(s, a, d, y_i, e_i) \]

**Select most relevant sentence**

\[ p(y = y_i | s, a, d) \propto e^{\tilde{u} \cdot \tilde{\phi}(y_i, s, a, d)} \]

**Predict sentence predicate structure**

\[ p(e_j | j, y, q) \propto e^{\tilde{u} \cdot \tilde{\psi}(e_j, j, y, q)} \]

**Predicted action value**
Parameter Estimation

**Objective:** Minimize *mean square error* between predicted utility \( Q(s, a, d) \) and observed utility \( R(s_\tau) \)

**Method:** Gradient descent — i.e., Backpropagation.
Experimental Domain

Game:
- Complex, stochastic turn-based strategy game Civilization II.
- Branching factor: $10^{20}$

Document:
- Official game manual of Civilization II

Text Statistics:
- Sentences: 2083
- Avg. sentence words: 16.7
- Vocabulary: 3638

Preface & Instruction Manual

At the start of the game, your civilization consists of a single band of wandering nomads. This is a settlers unit. Although settlers are capable of performing a variety of useful tasks, your first task is to move the settlers unit to a site that is suitable for the construction of your first city. Finding suitable locations in which to build cities, especially your first city, is one of the most important decisions you make in the
Experimental Setup

Game opponent:

- *Built-in AI of Game.*
- *Domain knowledge rich AI, built to challenge humans.*

Primary evaluation:

- *Games won within first 100 game steps.*
- *Averaged over 200 independent experiments.*
- *Avg. experiment runtime: 1.5 hours*

Secondary evaluation:

- *Full games won.*
- *Averaged over 100 independent experiments.*
- *Avg. experiment runtime: 4 hours*
% games won in 100 turns, averaged over 200 runs.

Full model: 53.7%

Built-in AI: 0%
Does Text Help?

- **Built-in AI**: 0%
- **Game only**: 17.3%
- **Full model**: 53.7%

**Linear Q fn. approximation, No text**

% games won in 100 turns, averaged over 200 runs.
Text vs. Representational Capacity

- **Built-in AI**: 0%
- **Game only**: 17.3%
- **Latent variable**: 26.1%
- **Full model**: 53.7%

% games won in 100 turns, averaged over 200 runs.

*Non-Linear Q fn. approximation, No text*
Linguistic Complexity vs. Performance Gain

- Built-in AI: 0%
- Game only: 17.3%
- Latent variable: 26.1%
- Sentence relevance: 46.7%
- Full model: 53.7%

% games won in 100 turns, averaged over 200 runs.
Results: Sentence Relevance

Problem: *Sentence relevance depends on game state.*

*States are game specific, and not known a priori!*

Solution: *Add known non-relevant sentences to text.*

*E.g., sentences from the Wall Street Journal corpus.*

Results: *71.8% sentence relevance accuracy...*  

*Surprisingly poor accuracy given game win rate!*
Results: Sentence Relevance
Results: Full Games

- Game only: 24.8%
- Latent variable: 31.5%
- Full model: 65.4%

Percentage games won, averaged over 100 runs
Outline

1. Step-by-step imperative instructions
   - Learning from control feedback

2. High-level strategy descriptions
   - Situational relevance
   - Incompleteness
   - Learning from control feedback

3. General descriptions of world dynamics
   - Abstractions
   - Situational relevance
   - Incompleteness
   - Learning from control feedback
Solving Hard Planning Tasks

**Objective:** Compute plan to achieve given goal

**Challenge:** Exponential search space

**Traditional solution:** Analyze domain structure to induce sub-goals

**Our goal:** Use precondition information from text to guide sub-goal induction
Solving Hard Planning Tasks

Minecraft: Virtual world allowing tool creation and complex construction.
Seeds planted in farmland will grow to become wheat which can be harvested.
Seeds planted in farmland will grow to become wheat which can be harvested.

**Preconditions**

- seeds $\rightarrow$ wheat
- farmland $\rightarrow$ wheat

**Plan**

1. pickup tool: shears
2. collect seeds from tallgrass using shears
3. pickup tool: hoe
4. plow land with hoe at (2,0) into farmland
5. plant seeds at coordinates (2,0)
6. fertilize seeds at (2,0) with bonemeal
7. wait for wheat to grow
8. pickup tool: shears
9. harvest wheat with shears at (2,0)
Learn model parameters from planning feedback

Model

Text → Log-linear model

Planning target goal → Log-linear model

Precondition relations

Sub-goal sequence

Low-level planner (FF) → Plan

Model precondition descriptions

Model object relations in world, and ground preconditions.

Learn model parameters from planning feedback
Experimental Domain

World:
Minecraft virtual world

Documents:
User authored wiki articles

Text Statistics:

- Sentences: 242
- Vocabulary: 979

Planning task Statistics:

- Tasks: 98
- Avg. plan length: 35

Pickaxes

Pickaxes are one of the most commonly used tools in the game, being required to mine all ores and many other types of blocks. Different qualities of pickaxe are required to successfully
## Results

<table>
<thead>
<tr>
<th>Method</th>
<th>% plans solved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low-level planner (FF)</td>
<td>40.8</td>
</tr>
<tr>
<td>No text</td>
<td>69.4</td>
</tr>
<tr>
<td><strong>Full model</strong></td>
<td><strong>80.2</strong></td>
</tr>
<tr>
<td>Manual text connections</td>
<td>84.7</td>
</tr>
</tbody>
</table>
Results: Text Analysis

![Graph showing the accuracy of text analysis models over learning iterations. The graph plots the prediction F-score against learning iterations. The model F-score curve (green line) shows an initial steep increase and then plateaus at around 0.7, while the SVM F-score curve (red dashed line) remains constant at 0.7.]
Related Work

• Instruction interpretation
  *Learned from manual supervision*

• Game playing
  *Action selection based only on game state*

• High-level planning
  *Based on analysis of world dynamics*
Contributions

• Learn language analysis from control feedback

• Improve control performance using text information