Multitask Learning

Motivating Example

- 4 tasks defined on eight bits $B_1$-$B_8$:

  Task 1 = $B_1 \oplus \text{Parity}(B_2 \oplus B_6)$
  Task 2 = $\Box B_1 \oplus \text{Parity}(B_2 \oplus B_6)$
  Task 3 = $B_1 \boxslash \text{Parity}(B_2 \oplus B_6)$
  Task 4 = $\Box B_1 \boxslash \text{Parity}(B_2 \oplus B_6)$

Motivating Example: STL & MTL

Motivating Example: Results
Motivating Example: **Why?**

**extra tasks:**
- add noise?
- change learning rate?
- reduce herd effect by differentiating hu’s?
- use excess net capacity?
- . . . ?
- similarity to main task helps hidden layer learn better representation?

---

**Autonomous Vehicle Navigation ANN**

---

**Multitask Learning for ALVINN**
Problem 1: 1D-ALVINN

- simulator developed by Pomerleau
- **main task: steering direction**
- 8 extra tasks:
  - 1 or 2 lanes
  - horizontal location of centerline
  - horizontal location of road center, left edge, right edge
  - intensity of centerline, road surface, burms

<table>
<thead>
<tr>
<th>TASK</th>
<th>STL 2hu</th>
<th>STL 4hu</th>
<th>STL 8hu</th>
<th>STL 16hu</th>
<th>MTL 16hu</th>
<th>%Change Best</th>
<th>%Change Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 or 2 Lanes</td>
<td>0.201</td>
<td>0.209</td>
<td>0.207</td>
<td>0.197</td>
<td>0.186</td>
<td>-12.40%</td>
<td>-21.50%</td>
</tr>
<tr>
<td>Left Edge</td>
<td>0.076</td>
<td>0.071</td>
<td>0.073</td>
<td>0.073</td>
<td>0.073</td>
<td>-13.10%</td>
<td>-13.30%</td>
</tr>
<tr>
<td>Right Edge</td>
<td>0.076</td>
<td>0.062</td>
<td>0.058</td>
<td>0.056</td>
<td>0.051</td>
<td>-19.00%</td>
<td>-19.00%</td>
</tr>
<tr>
<td>Line Center</td>
<td>0.153</td>
<td>0.152</td>
<td>0.152</td>
<td>0.151</td>
<td>0.151</td>
<td>-0.70%</td>
<td>-0.80%</td>
</tr>
<tr>
<td>Road Center</td>
<td>0.038</td>
<td>0.031</td>
<td>0.039</td>
<td>0.042</td>
<td>0.034</td>
<td>-12.80%</td>
<td>-12.80%</td>
</tr>
<tr>
<td>Road Greylevel</td>
<td>0.035</td>
<td>0.055</td>
<td>0.055</td>
<td>0.054</td>
<td>0.036</td>
<td>-30.30%</td>
<td>-30.30%</td>
</tr>
<tr>
<td>Edge Greylevel</td>
<td>0.038</td>
<td>0.039</td>
<td>0.038</td>
<td>0.038</td>
<td>0.038</td>
<td>0.00%</td>
<td>0.00%</td>
</tr>
<tr>
<td>Line Greylevel</td>
<td>0.054</td>
<td>0.054</td>
<td>0.054</td>
<td>0.054</td>
<td>0.054</td>
<td>0.00%</td>
<td>0.00%</td>
</tr>
<tr>
<td>Steering</td>
<td>0.083</td>
<td>0.081</td>
<td>0.087</td>
<td>0.072</td>
<td>0.056</td>
<td>15.90%</td>
<td>-27.70%</td>
</tr>
</tbody>
</table>

Problem 2: 1D-Doors

- color camera on Xavier robot
- **main tasks: doorknob location and door type**
- 8 extra tasks (training signals collected by mouse):
  - doorway width
  - location of doorway center
  - location of left jamb, right jamb
  - location of left and right edges of door

1D-Doors: Results

- 20% more accurate doorknob location
- 35% more accurate doorway width
Predicting Pneumonia Risk

Pneumonia: Hospital Labs as Inputs

Pneumonia #1: Medis
Pneumonia #1: Results

-10.8% -11.8% -6.2% -6.9% -5.7%

Use imputed values for missing lab tests as extra inputs?

Pneumonia #1: Feature Nets

Feature Nets vs. MTL
Pneumonia #2: PORT

- 10X fewer cases (2286 patients)
- 10X more input features (200 feats)
- missing features (5% overall, up to 50%)
- main task: dire outcome
- 30 extra tasks currently available
  - dire outcome disjuncts (death, ICU, cardio, ...)
  - length of stay in hospital
  - cost of hospitalization
  - etiology (gramnegative, grampositive, ...)
  - ...

MTL reduces error >10%

Related?

- related \( \mathcal{Y} \) helps learning (e.g., copy task)
- helps learning \( \mathcal{Y} \) related (e.g., noise task)
- related \( \mathcal{Y} \) correlated (e.g., A+B, A-B)

Two tasks are MTL/BP related if there is correlation (positive or negative) between the training signals of one and the hidden layer representation learned for the other

120 Synthetic Tasks

- backprop net not told how tasks are related, but ...
- 120 Peaks Functions: A,B,C,D,E,F \[ \{0.0,1.0\} \)
  - \( P_{001} = \text{If } (A > 0.5) \text{ Then } B \text{ Else } C \)
  - \( P_{002} = \text{If } (A > 0.5) \text{ Then } B \text{ Else } D \)
  - \( P_{014} = \text{If } (A > 0.5) \text{ Then } E \text{ Else } C \)
  - \( P_{024} = \text{If } (B > 0.5) \text{ Then } A \text{ Else } F \)
  - \( P_{120} = \text{If } (F > 0.5) \text{ Then } E \text{ Else } D \)
Peaks Functions: Results

MTL nets *cluster* tasks by *function*
Heuristics: When to use MTL?

- using future to predict present
- time series
- disjunctive/conjunctive tasks
- multiple error metric
- quantized or stochastic tasks
- focus of attention
- sequential transfer
- different data distributions
- hierarchical tasks
- some input features work better as outputs

Multiple Tasks Occur Naturally

- Mitchell’s Calendar Apprentice (CAP)
  - time-of-day (9:00am, 9:30am, ...)
  - day-of-week (M, T, W, ...)
  - duration (30min, 60min, ...)
  - location (Tom’s office, Dean’s office, 5409, ...)

Using Future to Predict Present

- medical domains
- autonomous vehicles and robots
- time series
  - stock market
  - economic forecasting
  - weather prediction
  - spatial series
- many more

Disjunctive/Conjunctive Tasks

\[ \text{DireOutcome} = \text{ICU} \lor \text{Complication} \lor \text{Death} \]
Focus of Attention

- 1D-ALVNN:
  - centerline
  - left and right edges of road

Removing centerlines from 1D-ALVNN images hurts MTL accuracy more than STL accuracy.

Different Data Distributions

- Hospital 1: 50 cases, rural (Green Acres)
- Hospital 2: 500 cases, urban (Des Moines)
- Hospital 3: 1000 cases, elderly suburbs (Florida)
- Hospital 4: 5000 cases, young urban (LA, SF)

Some Inputs are Better as Outputs

- MainTask = Sigmoid(A)+Sigmoid(B)
- A, B ∈ (−5.0, +5.0)
- Inputs A and B coded via 10-bit binary code

Some Inputs are Better as Outputs

- MainTask = Sigmoid(A)+Sigmoid(B)
- Extra Features:
  - EF1 = Sigmoid(A) + * Noise
  - EF2 = Sigmoid(B) + * Noise
  - where ∈ (0.0, 10.0), Noise ∈ (−1.0, 1.0)
Inputs Better as Outputs: Results

Some Inputs Better as Outputs

Making MTL/Backprop Better

- Better training algorithm:
  - learning rate optimization
- Better architectures:
  - private hidden layers (overfitting in hidden unit space)
  - using features as both inputs and outputs
  - combining MTL with Feature Nets

Private Hidden Layers

- many tasks: need many hidden units
- many hidden units: “hidden unit selection problem”
- allow sharing, but without too many hidden units?
Features as Both Inputs & Outputs

- some features help when used as inputs
- some of those also help when used as outputs
- get both benefits in one net?

MTL in K-Nearest Neighbor

- Most learning methods can MTL:
  - shared representation
  - combine performance of extra tasks
  - control the effect of extra tasks

- MTL in K-Nearest Neighbor:
  - shared rep: distance metric
  - $\text{MTL Perf} = (1-\lambda)\text{Main Perf} + \lambda \text{Extra Perf}$

MTL/KNN for Pneumonia #1
Psychological Plausibility

Related Work
- Sejnowski, Rosenberg [1986]: NETtalk
- Pratt, Mostow [1991-94]: serial transfer in bp nets
- Suddarth, Kergiosen [1990]: 1st MTL in bp nets
- Abu-Mostafa [1990-95]: catalytic hints
- Abu-Mostafa, Baxter [92,95]: transfer PAC models
- Dietterich, Hild, Bakiri [90,95]: bp vs. ID3
- Pomerleau, Baluja: other uses of hidden layers
- Munro [1996]: extra tasks to decorrelate experts
- Breiman [1995]: Curds & Whey
- de Sa [1995]: minimizing disagreement
- Thrun, Mitchell [1994,96]: EBNN
- O’Sullivan, Mitchell [now]: EBNN+MTL+Robot

MTL vs. EBNN on Robot Problem

Parallel vs. Serial Transfer
- all information is in training signals
- information useful to other tasks can be lost training on tasks one at a time
- if we train on extra tasks first, how can we optimize what is learned to help the main task most
- tasks often benefit each other mutually
- parallel training allows related tasks to see the entire trajectory of other task learning
Summary/Contributions

- **focus on main** task improves performance
- >15 problem types where MTL is applicable:
  - using the future to predict the present
  - multiple metrics
  - focus of attention
  - different data populations
  - using inputs as extra tasks
  - ... (at least 10 more)
  
  *most real-world problems fit one of these*

- applied MTL to a dozen problems, some not created for MTL
  - MTL helps most of the time
  - benefits range from 5%-40%
- ways to improve MTL/Backprop
  - learning rate optimization
  - private hidden layers
  - MTL Feature Nets
- MTL nets do unsupervised clustering
- algs for MTL kNN and MTL Decision Trees

Future MTL Work

- output selection
- scale to 1000’s of extra tasks
- compare to Bayes Nets
- learning rate optimization

Theoretical Models of Parallel Xfer

- **PAC** models based on VC-dim or MDL
  - unreasonable assumptions
    + fixed size hidden layers
    + all tasks generated by one hidden layer
    + backprop is ideal search procedure
  - predictions do not fit observations
    + have to add hidden units
  - main problems:
    + can’t take behavior of backprop into account
    + not enough is known about capacity of backprop nets
Learning Rate Optimization

- optimize learning rates of extra tasks
- goal is maximize generalization of main task
- ignore performance of extra tasks
- expensive!

- performance on extra tasks improves 9%!

MTL Feature Nets

Acknowledgements

- advisors: Mitchell & Simon
- committee: Pomerleau & Dietterich
- CEHC: Cooper, Fine, Buchanan, et al.
- co-authors: Baluja, de Sa, Freitag
- robot Xavier: O’Sullivan, Simmons
- discussion: Fahlman, Moore, Touretzky
- funding: NSF, ARPA, DEC, CEHC, JPRC
- SCS/CMU: a great place to do research
- spouse: Diane