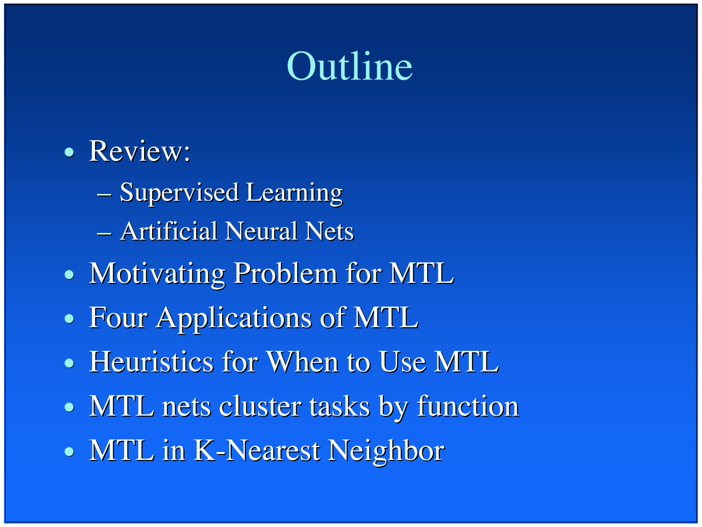
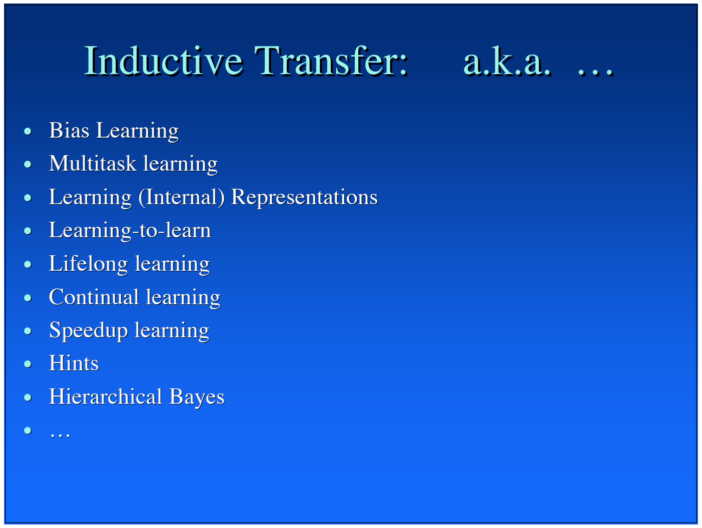
Inductive Transfer and Multitask Learning



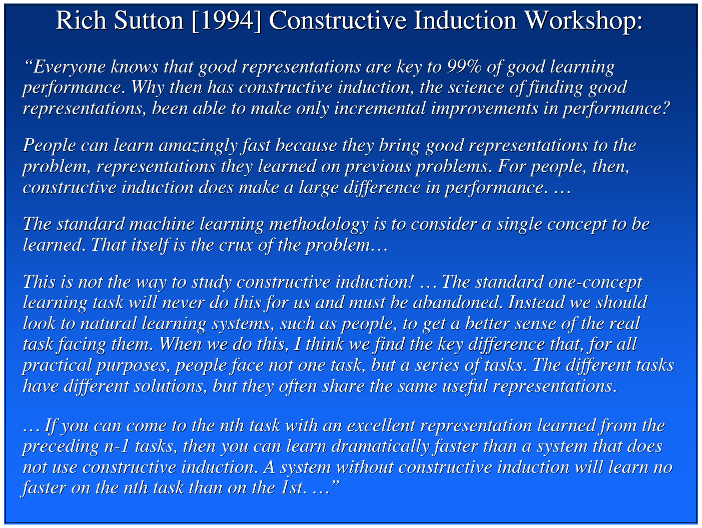
Outline

- Review:
  - Supervised Learning
  - Artificial Neural Nets
- Motivating Problem for MTL
- Four Applications of MTL
- Heuristics for When to Use MTL
- MTL nets cluster tasks by function
- MTL in K-Nearest Neighbor



Inductive Transfer: a.k.a. …

- Bias Learning
- Multitask learning
- Learning (Internal) Representations
- Learning-to-learn
- Lifelong learning
- Continual learning
- Speedup learning
- Hints
- Hierarchical Bayes
- …



Rich Sutton [1994] Constructive Induction Workshop:

"Everyone knows that good representations are key to 99% of good learning performance. Why then has constructive induction, the science of finding good representations, been able to make only incremental improvements in performance?

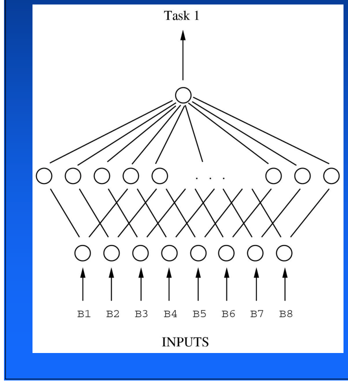
People can learn amazingly fast because they bring good representations to the problem, representations they learned on previous problems. For people, then, constructive induction does make a large difference in performance. …

The standard machine learning methodology is to consider a single concept to be learned. That itself is the crux of the problem…

This is not the way to study constructive induction! … The standard one-concept learning task will never do this for us and must be abandoned. Instead we should look to natural learning systems, such as people, to get a better sense of the real task facing them. When we do this, I think we find the key difference that, for all practical purposes, people face not one task, but a series of tasks. The different tasks have different solutions, but they often share the same useful representations.

… If you can come to the nth task with an excellent representation learned from the preceding n-1 tasks, then you can learn dramatically faster than a system that does not use constructive induction. A system without constructive induction will learn no faster on the nth task than on the 1st. …"

What are Artificial Neural Nets?



- supervised learning
- generalized nonlinear regression method
- continuous: trained with gradient descent
- hidden layer is learned features
- propositional learning (not first order)
- perform well in practice

Motivating Example

- 4 tasks defined on eight bits B_1-B_8:

$$\text{Task 1} = \quad B_1 \quad \text{Parity}(B_2 \square B_6)$$

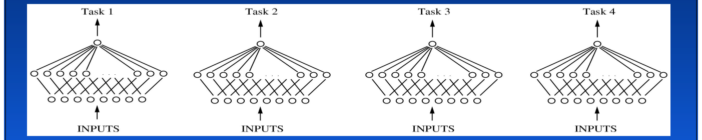
$$\text{Task 2} = \square B_1 \quad \text{Parity}(B_2 \square B_6)$$

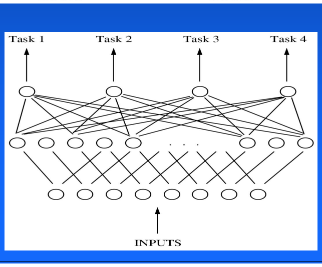
$$\text{Task 3} = \quad B_1 \square \text{Parity}(B_2 \square B_6)$$

$$\text{Task 4} = \square B_1 \square \text{Parity}(B_2 \square B_6)$$

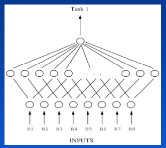
- all tasks ignore input bits B_7-B_8

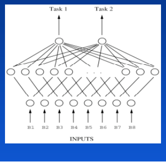
Motivating Example: STL & MTL

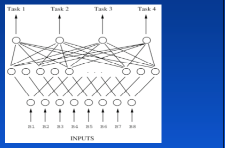


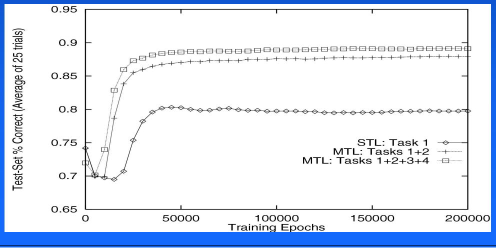


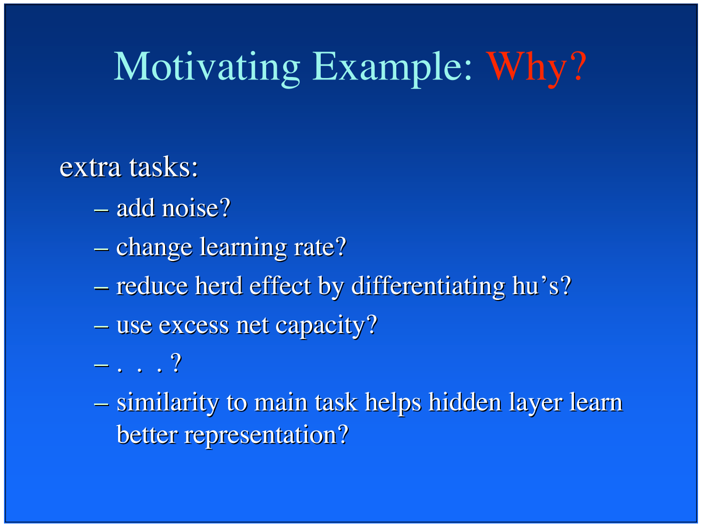
Motivating Example: Results







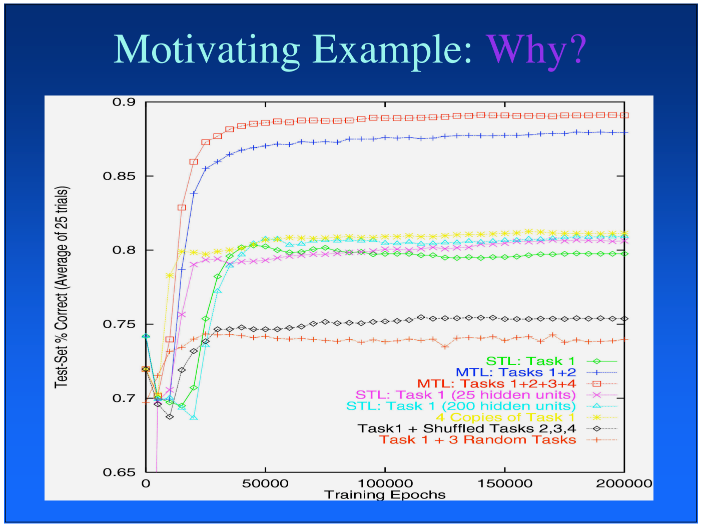




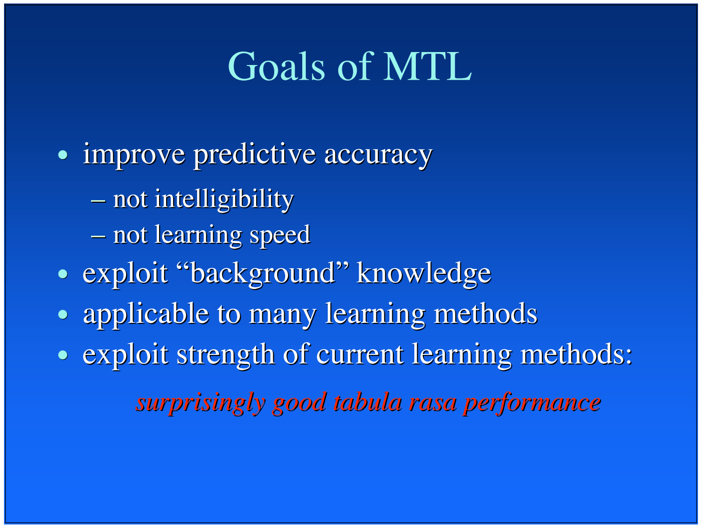
Motivating Example: Why?

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?



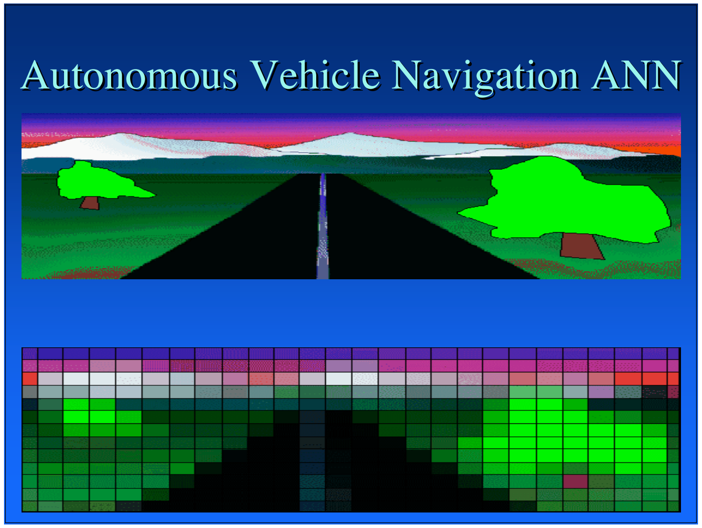
Motivating Example: Why?



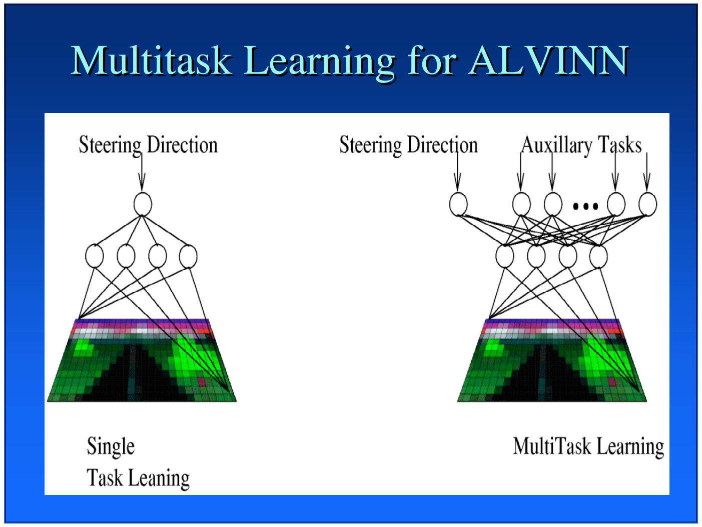
Goals of MTL

- improve predictive accuracy
  - not intelligibility
  - not learning speed
- exploit "background" knowledge
- applicable to many learning methods
- exploit strength of current learning methods:

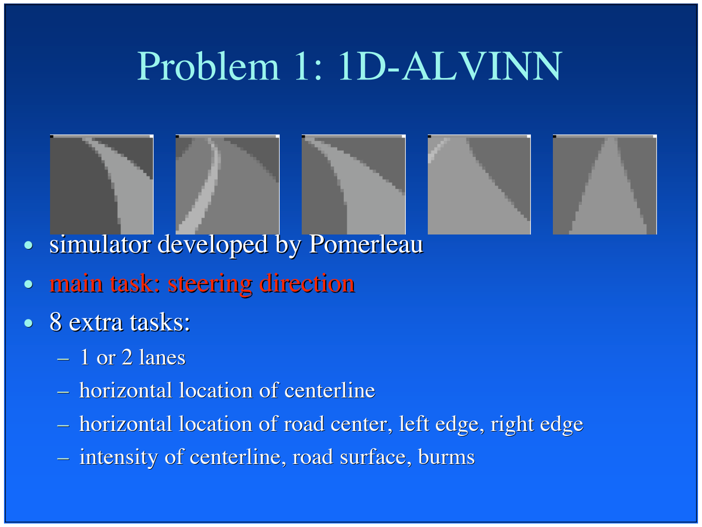
surprisingly good tabula rasa performance



Autonomous Vehicle Navigation ANN

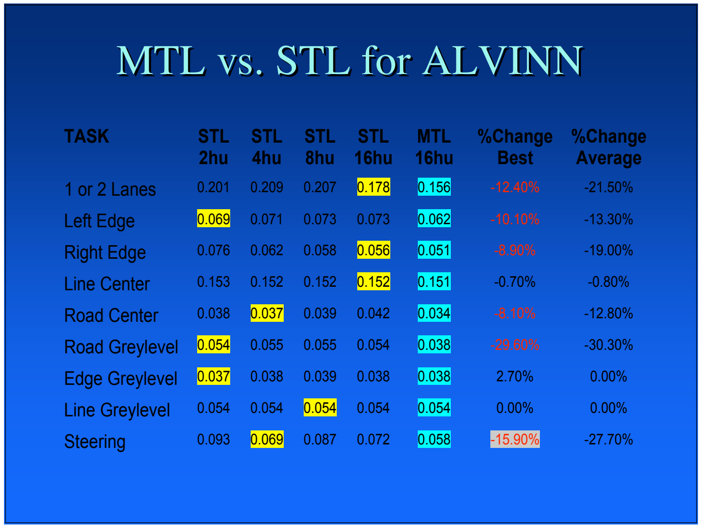


Multitask Learning for ALVINN



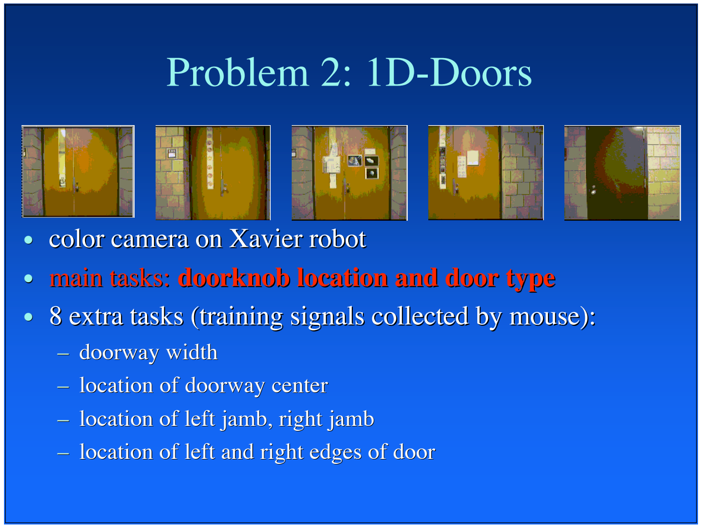
Problem 1: 1D-ALVINN

- 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



MTL vs. STL for ALVINN

TASK	STL 2hu	STL 4hu	STL 8hu	STL 16hu	MTL 16hu	%Change Best	%Change Average
1 or 2 Lanes	0.201	0.209	0.207	0.178	0.156	-12.40%	-21.50%
Left Edge	0.069	0.071	0.073	0.073	0.062	-10.10%	-13.30%
Right Edge	0.076	0.062	0.058	0.056	0.051	-8.90%	-19.00%
Line Center	0.153	0.152	0.152	0.152	0.151	-0.70%	-0.80%
Road Center	0.038	0.037	0.039	0.042	0.034	-8.10%	-12.80%
Road Greylevel	0.054	0.055	0.055	0.054	0.038	-29.60%	-30.30%
Edge Greylevel	0.037	0.038	0.039	0.038	0.038	2.70%	0.00%
Line Greylevel	0.054	0.054	0.054	0.054	0.054	0.00%	0.00%
Steering	0.093	0.069	0.087	0.072	0.058	-15.90%	-27.70%



Problem 2: 1D-Doors

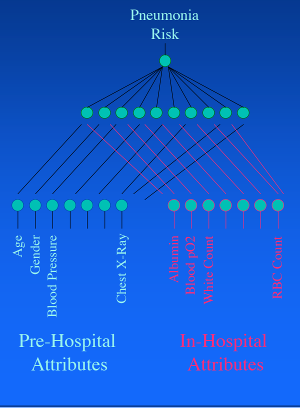
- color camera on Xavier robot
- main tasks: **doorknob location and door type**
- 8 extra tasks (training signals collected by mouse):
  - doorway width
  - location of doorway center
  - location of left jamb, right jamb
  - location of left and right edges of door

1D-Doors: Results

20% more accurate doorknob location

35% more accurate doorway width

Predicting Pneumonia Risk



Pneumonia Risk

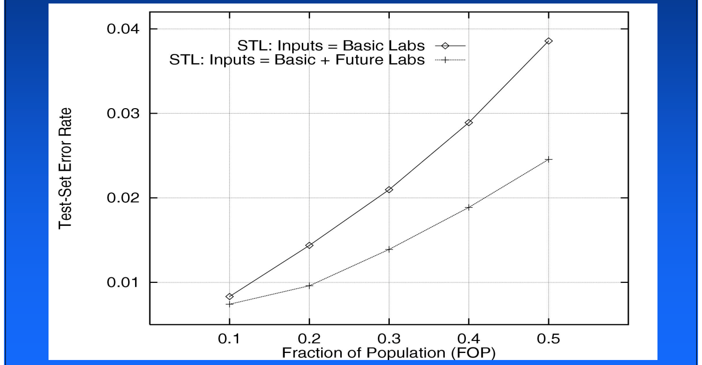
Age
Gender
Blood Pressure
Chest X-Ray

Albumin
Blood pO2
White Count
RBC Count

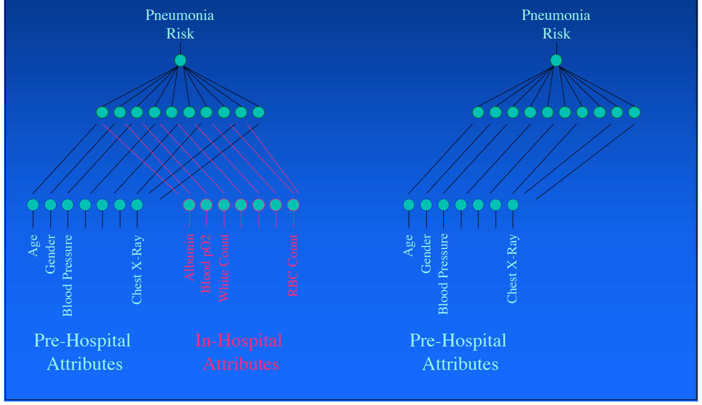
Pre-Hospital Attributes

In-Hospital Attributes

Pneumonia: Hospital Labs as Inputs



Predicting Pneumonia Risk



Pneumonia Risk

Age
Gender
Blood Pressure
Chest X-Ray

Albumin
Blood pO2
White Count
RBC Count

Pre-Hospital Attributes

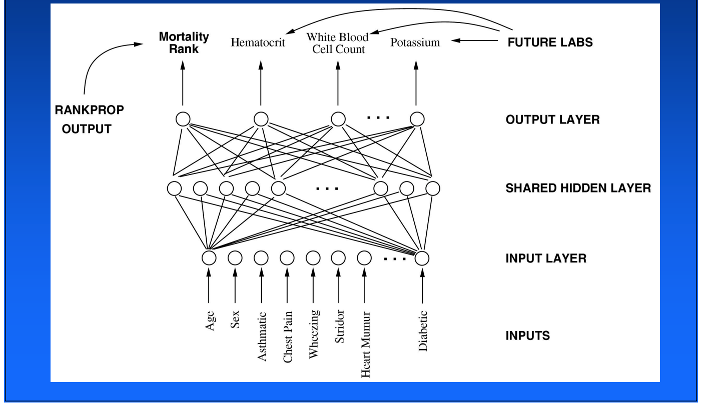
In-Hospital Attributes

Pneumonia Risk

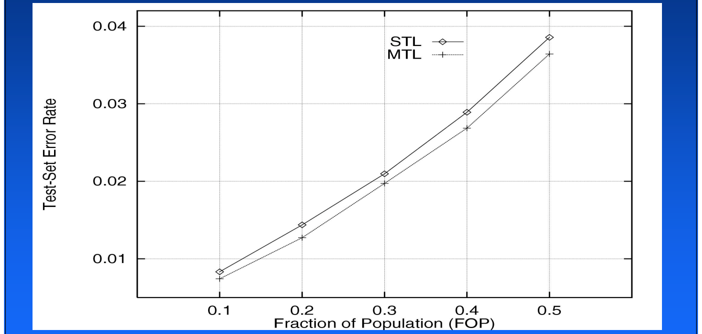
Age
Gender
Blood Pressure
Chest X-Ray

Pre-Hospital Attributes

Pneumonia #1: Medis



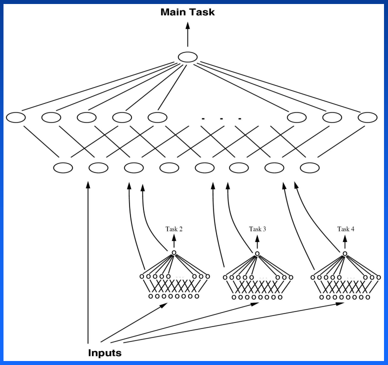
Pneumonia #1: Results

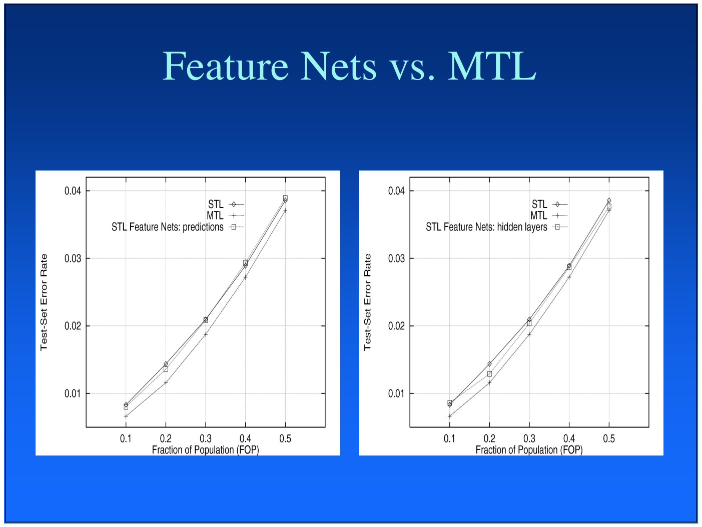


-10.8% -11.8% -6.2% -6.9% -5.7%

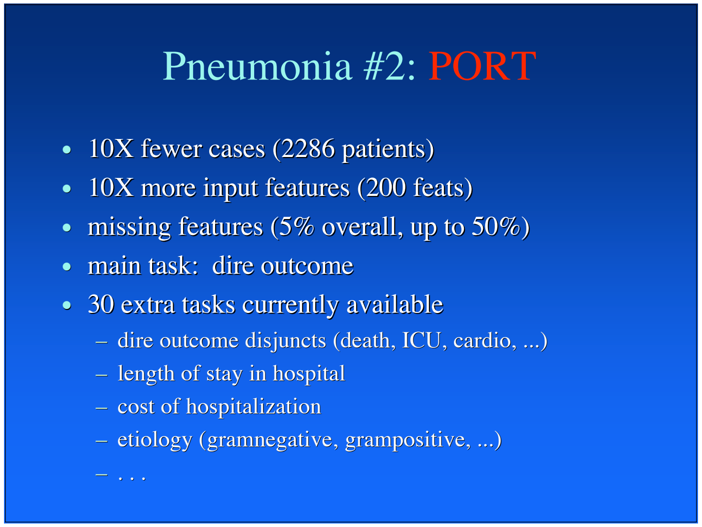
Use imputed values for missing lab tests as extra *inputs*?

Pneumonia #1: Feature Nets



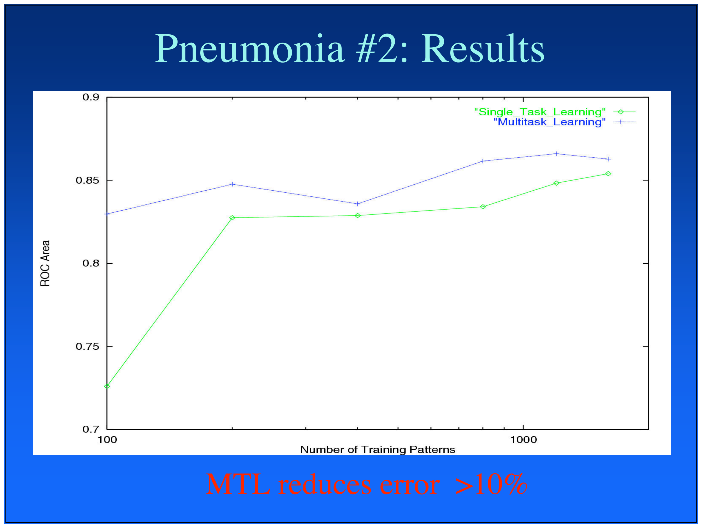


Feature Nets vs. MTL



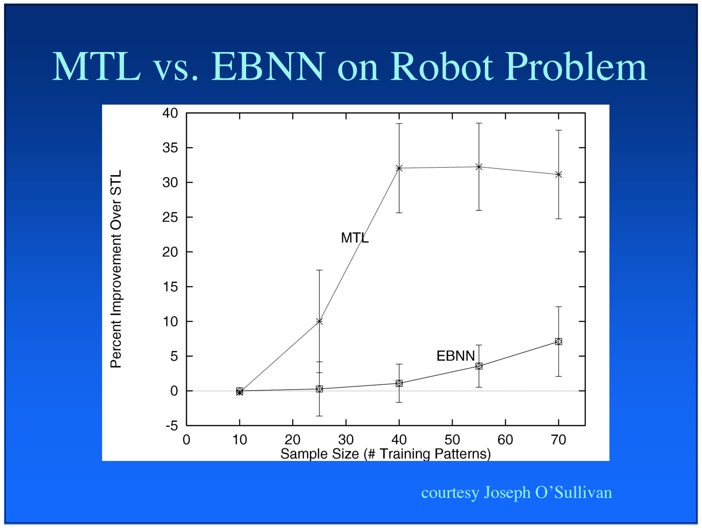
Pneumonia #2: PORT

- 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, ...)
  - . . .



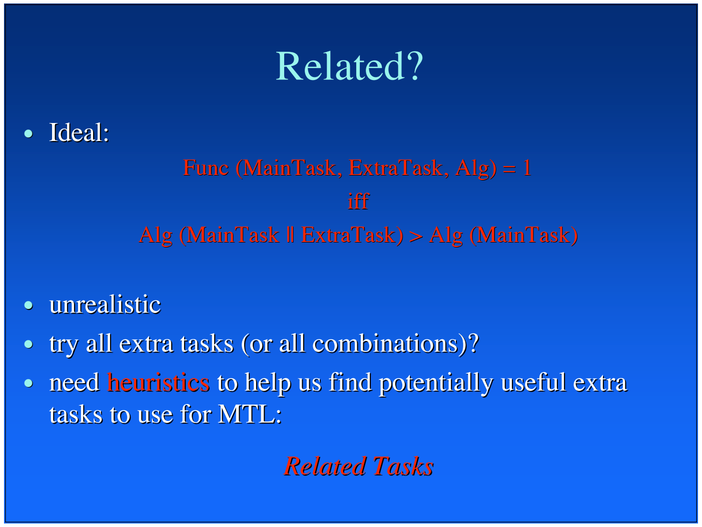
Pneumonia #2: Results

MTL reduces error >10%



MTL vs. EBNN on Robot Problem

courtesy Joseph O'Sullivan



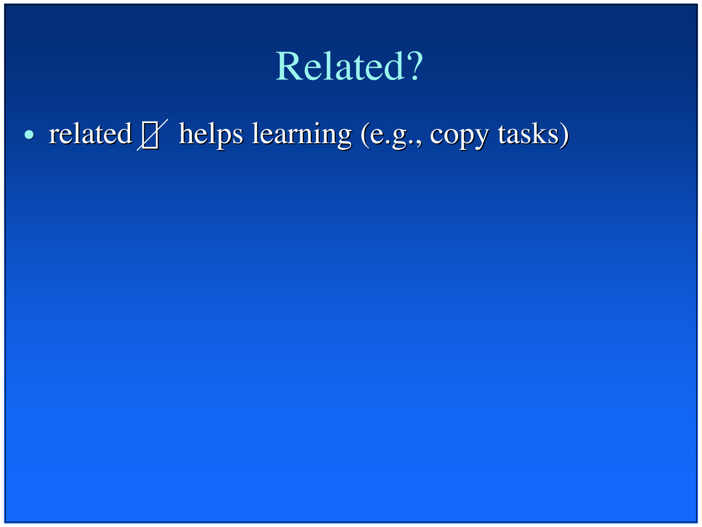
Related?

- Ideal:

  Func (MainTask, ExtraTask, Alg) = 1
  iff
  Alg (MainTask || ExtraTask) > Alg (MainTask)

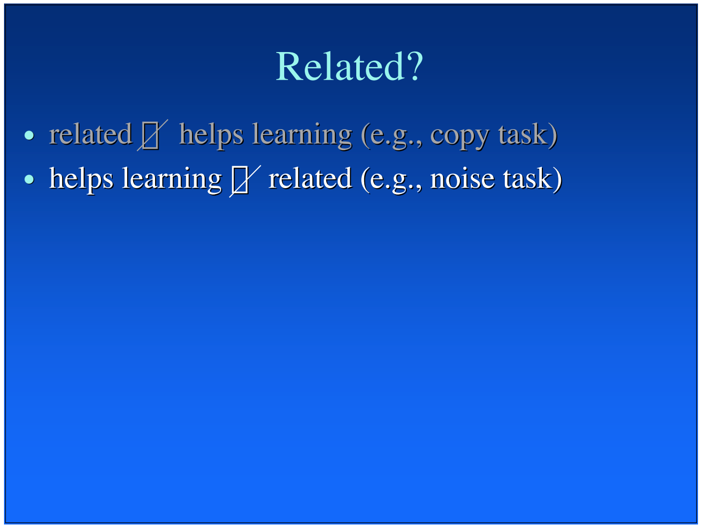
- unrealistic
- try all extra tasks (or all combinations)?
- need heuristics to help us find potentially useful extra tasks to use for MTL:

  Related Tasks



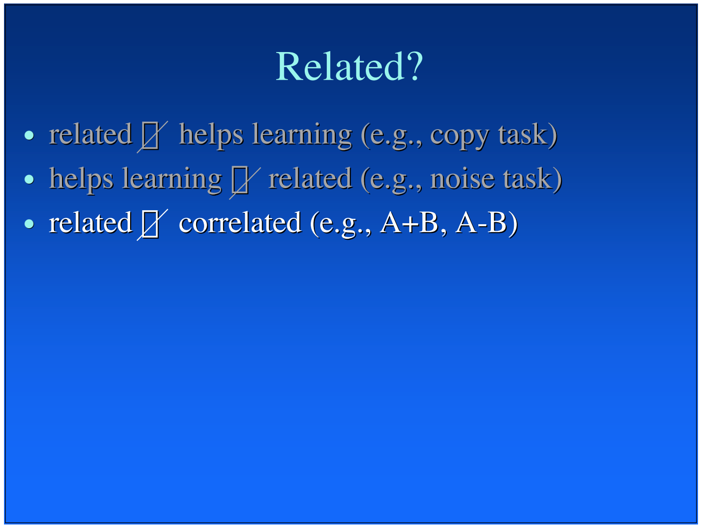
Related?

- related $\not\Rightarrow$ helps learning (e.g., copy tasks)



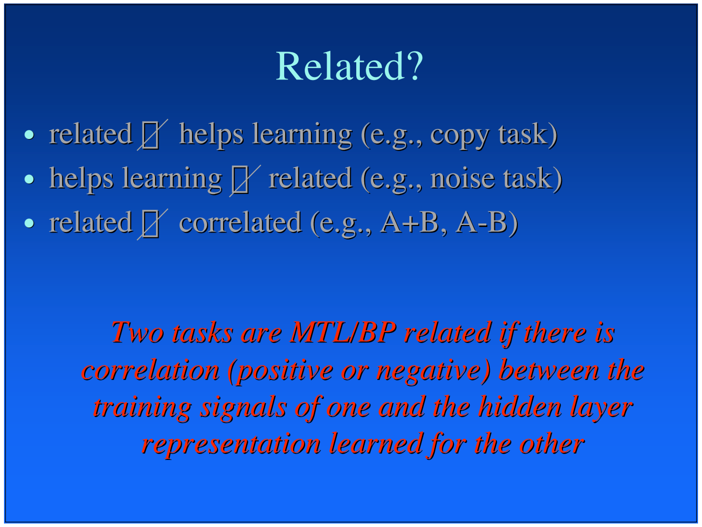
Related?

- related $\not\Rightarrow$ helps learning (e.g., copy task)
- helps learning $\not\Rightarrow$ related (e.g., noise task)



Related?

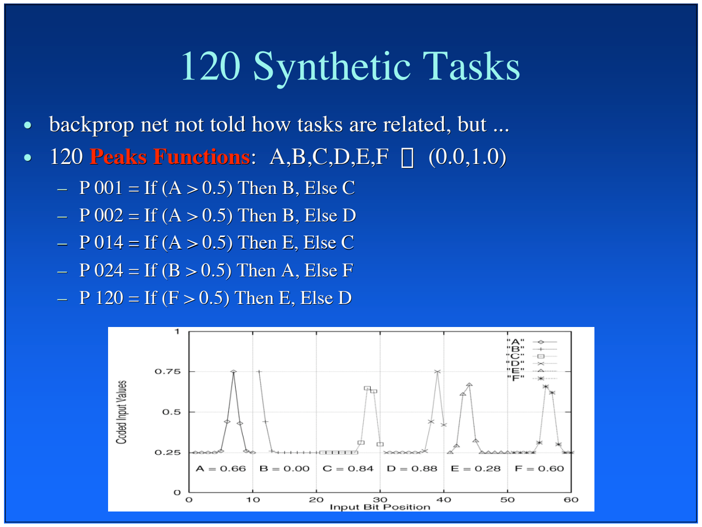
- related $\not\Rightarrow$ helps learning (e.g., copy task)
- helps learning $\not\Rightarrow$ related (e.g., noise task)
- related $\not\Rightarrow$ correlated (e.g., A+B, A-B)



Related?

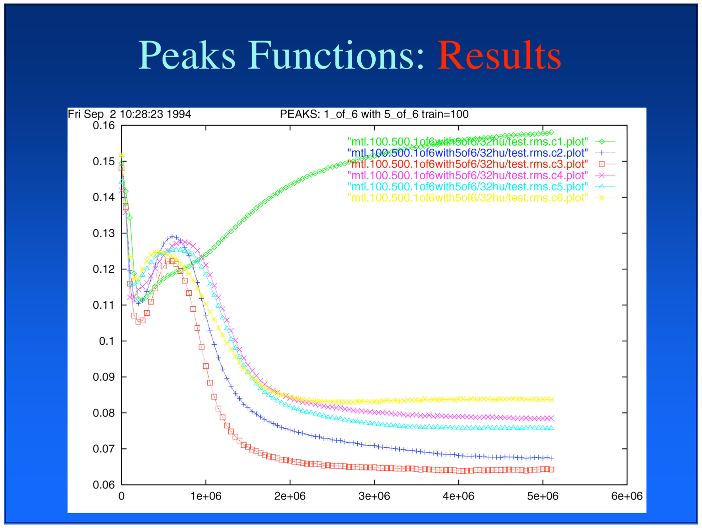
- related ⇏ helps learning (e.g., copy task)
- helps learning ⇏ related (e.g., noise task)
- related ⇏ correlated (e.g., A+B, A-B)

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

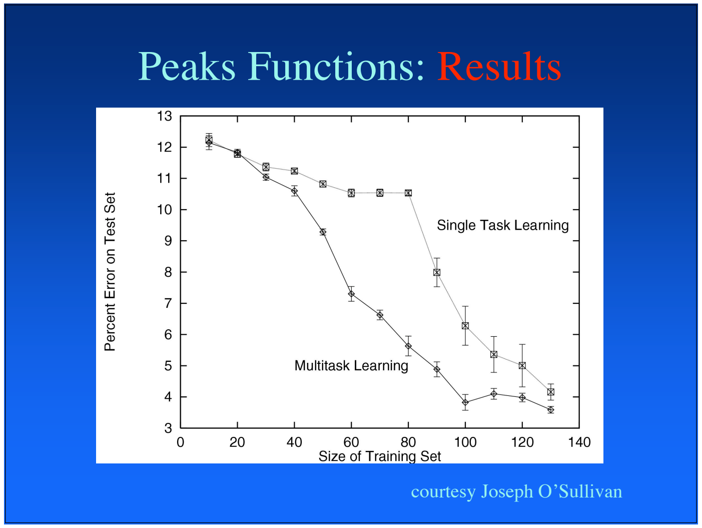


120 Synthetic Tasks

- backprop net not told how tasks are related, but ...
- 120 **Peaks Functions**: A,B,C,D,E,F ∈ (0.0,1.0)
  - P 001 = If (A > 0.5) Then B, Else C
  - P 002 = If (A > 0.5) Then B, Else D
  - P 014 = If (A > 0.5) Then E, Else C
  - P 024 = If (B > 0.5) Then A, Else F
  - P 120 = If (F > 0.5) Then E, Else D



Peaks Functions: Results

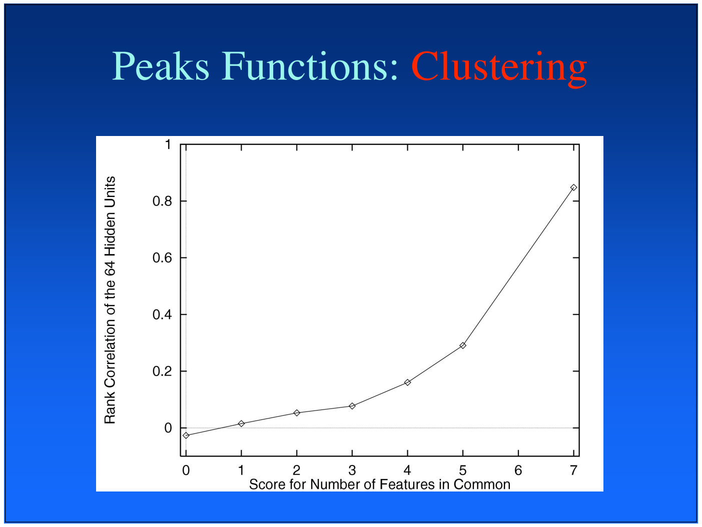


Peaks Functions: Results

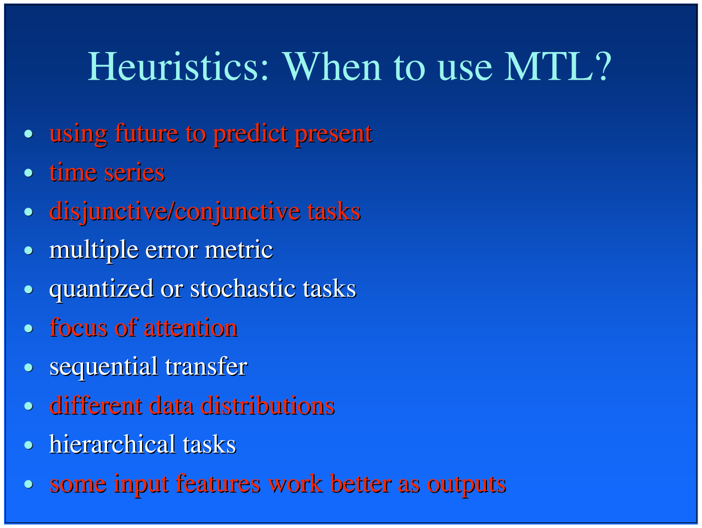
courtesy Joseph O'Sullivan

MTL nets cluster tasks

by *function*

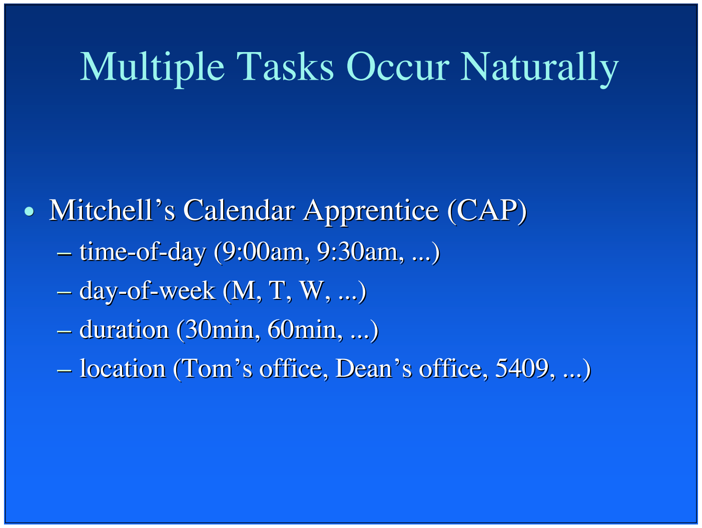


Peaks Functions: Clustering



Heuristics: When to use MTL?

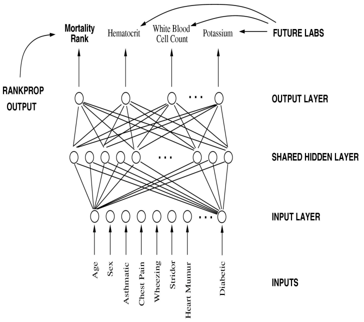
- using future to predict present
- time series
- disjunctive/conjunctive tasks
- multiple error metric
- quantized or stochastic tasks
- focus of attention
- sequential transfer
- different data distributions
- hierarchical tasks
- some input features work better as outputs



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, ...)

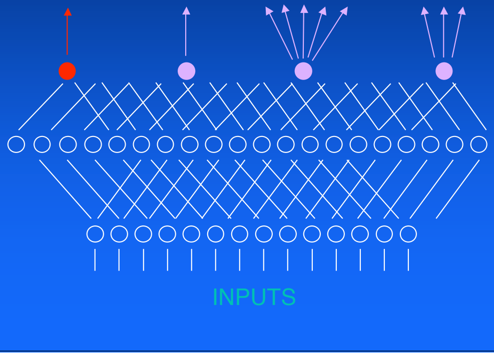
Using Future to Predict Present



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

Disjunctive/Conjunctive Tasks

DireOutcome = ICU v Complication v Death

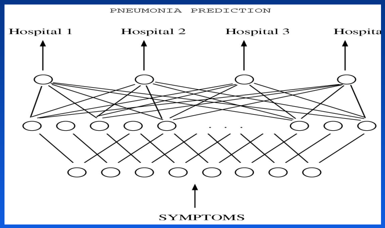


Focus of Attention

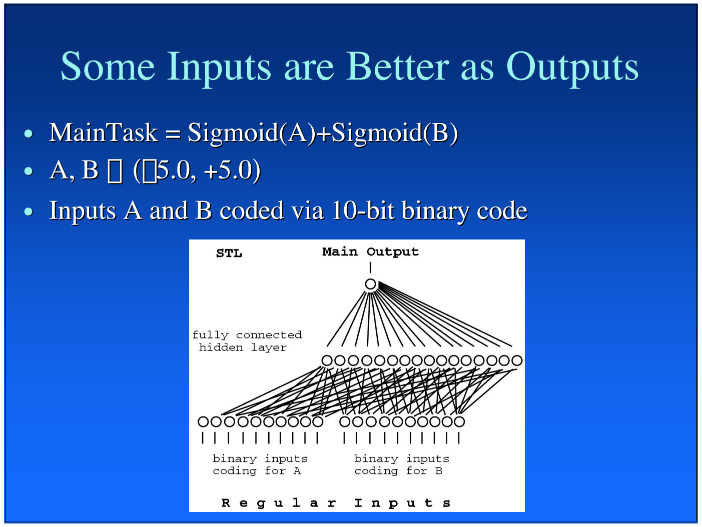
- 1D-ALVINN:
  - centerline
  - left and right edges of road

removing centerlines from 1D-ALVINN images hurts MTL accuracy more than STL accuracy

Different Data Distributions



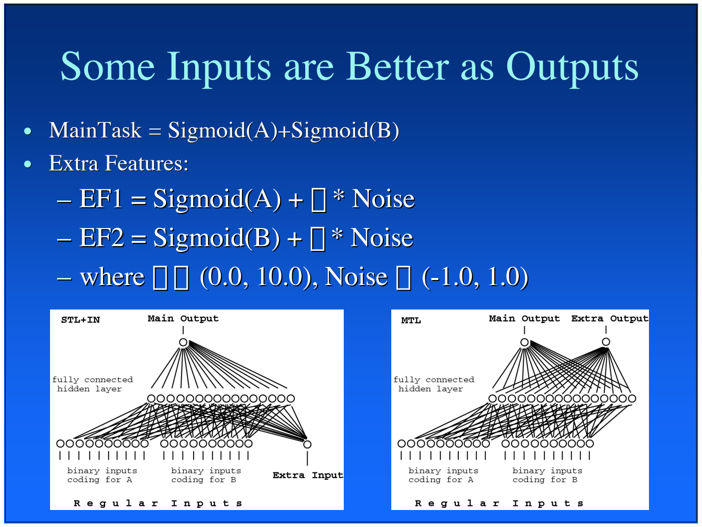
- Hospital 1: 50 cases, rural (Ithaca or Williamstown)
- Hospital 2: 500 cases, urban (Des Moines)
- Hospital 3: 1000 cases, elderly suburbs (Florida)
- Hospital 4: 5000 cases, young urban (LA,SF)



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

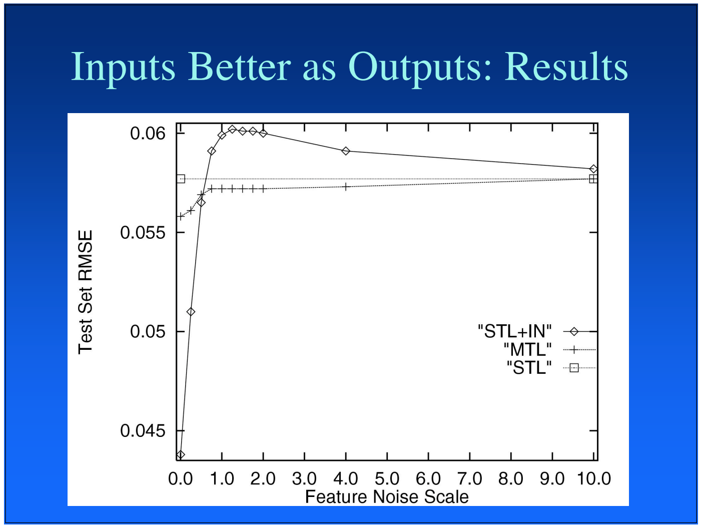
STL
Main Output
fully connected hidden layer
binary inputs coding for A
binary inputs coding for B
Regular Inputs



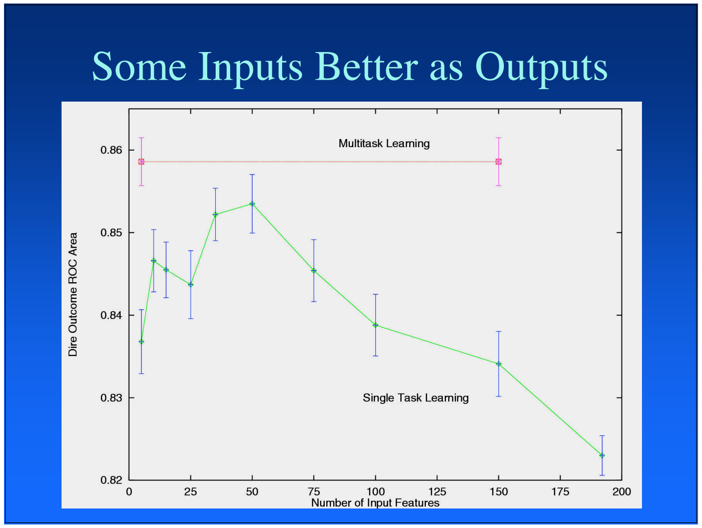
Some Inputs are Better as Outputs

- 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)

STL+IN
Main Output
fully connected hidden layer
binary inputs coding for A
binary inputs coding for B
Extra Input
Regular Inputs
MTL
Main Output
Extra Output
fully connected hidden layer
binary inputs coding for A
binary inputs coding for B
Regular Inputs



Inputs Better as Outputs: Results



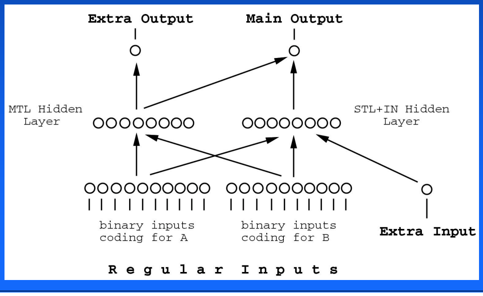
Some Inputs Better as Outputs

Extra Task (Output) Selection?

- Can’t try all possible combinations of inputs and outputs
- Even forward stepwise selection is expensive
- Bagging over inputs/outputs, perhaps with a form of bayesian weighting to reduce effect of bad models?
- Feature boosting finds combinations of input attributes that yield robust performance
- Is there a way to combine boosting with output task selection?

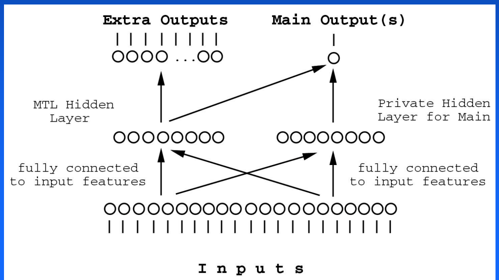
Features as Both Inputs & Outputs

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

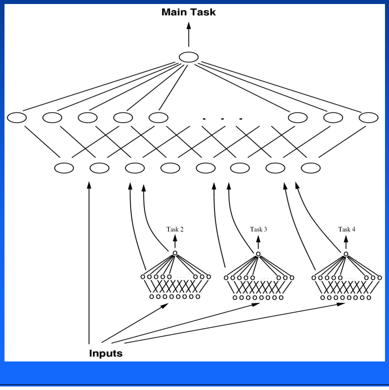


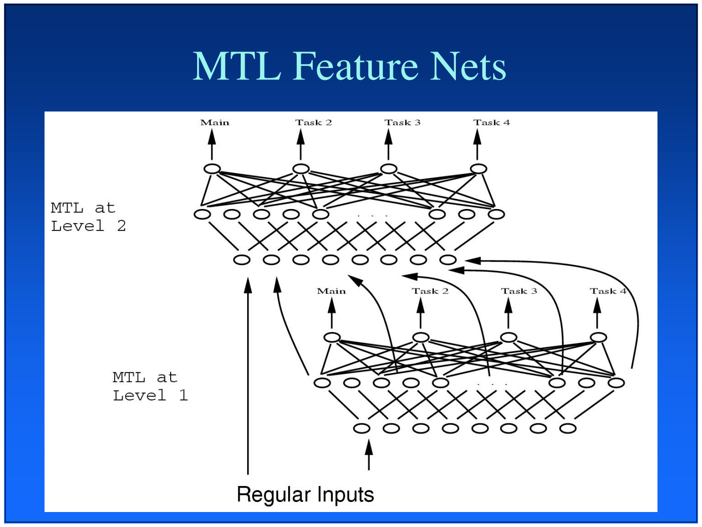
Private Hidden Layers

- many tasks: need many hidden units
- many hidden units: “hidden unit selection problem”
- allow sharing, but without too many hidden units?

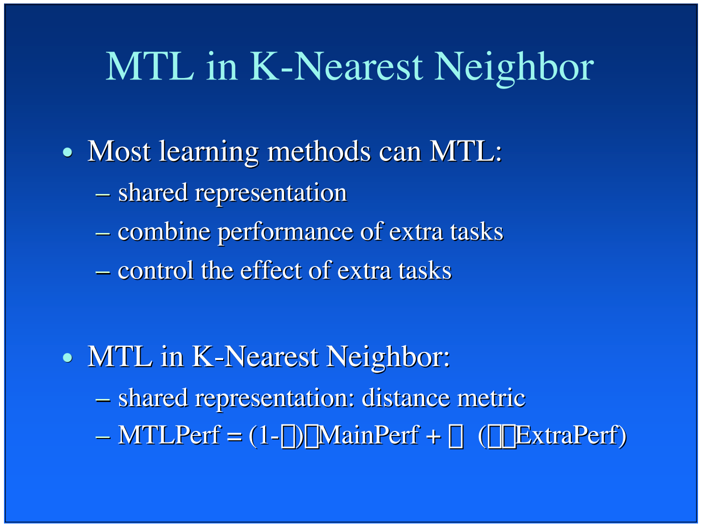

Extra Outputs
Main Output(s)
MTL Hidden Layer
Private Hidden Layer for Main
fully connected to input features
fully connected to input features
Inputs

Pneumonia #1: Feature Nets



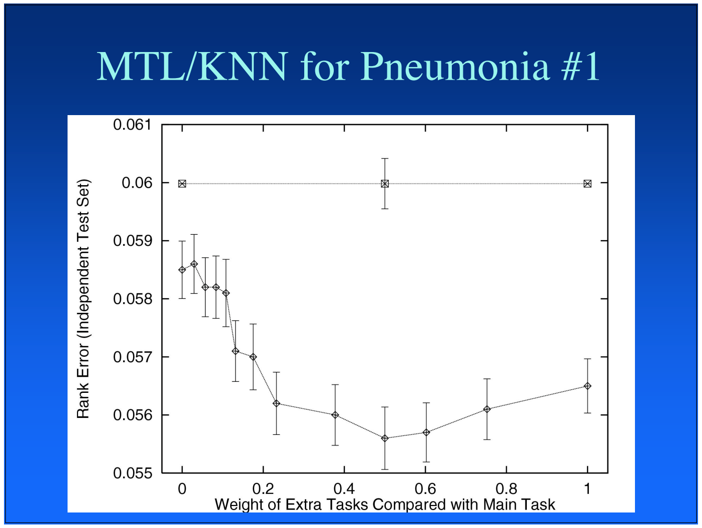


MTL Feature Nets

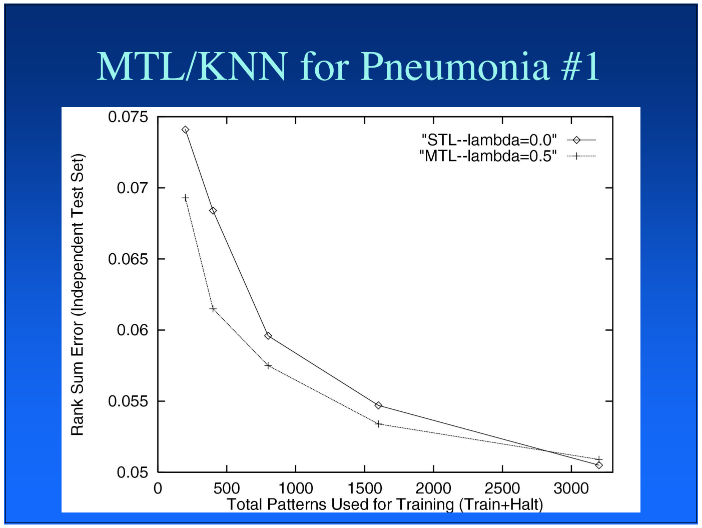


MTL in K-Nearest Neighbor

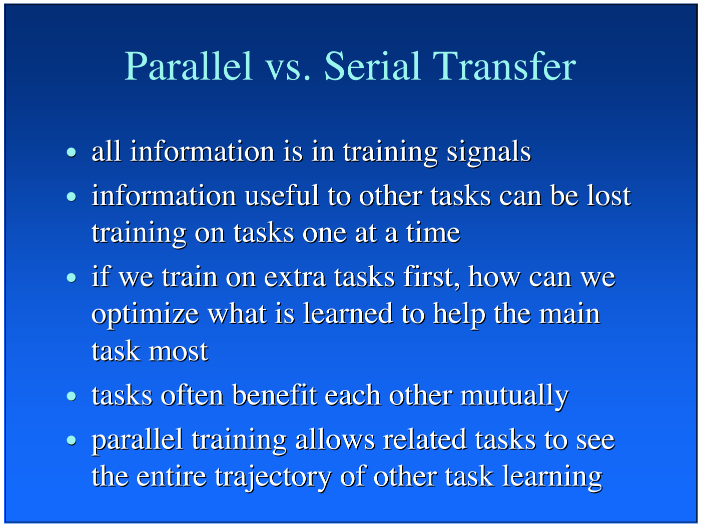
- Most learning methods can MTL:
  - – shared representation
  - – combine performance of extra tasks
  - – control the effect of extra tasks
- MTL in K-Nearest Neighbor:
  - – shared representation: distance metric
  - – MTLPerf = (1-□)□MainPerf + □ (□□ExtraPerf)



MTL/KNN for Pneumonia #1

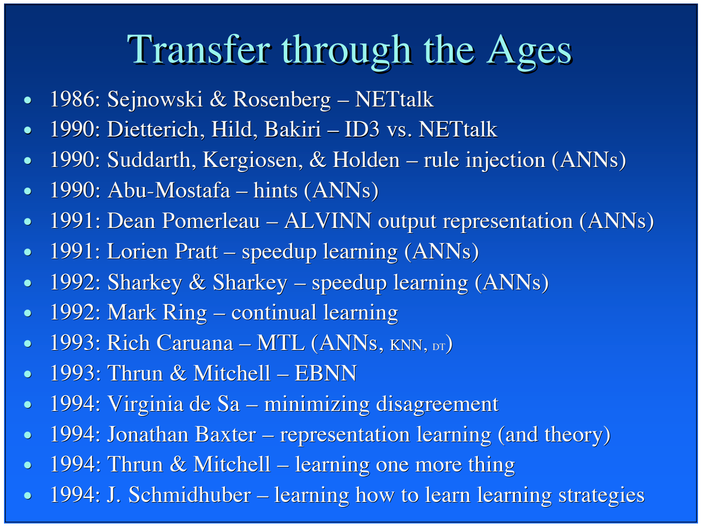


MTL/KNN for Pneumonia #1



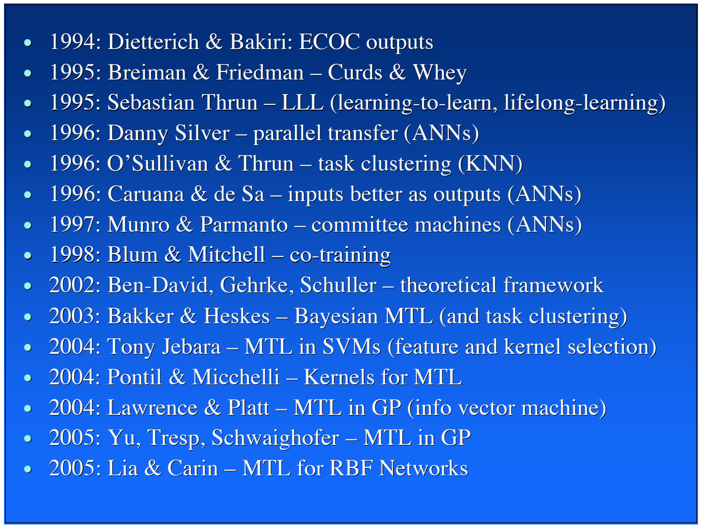
Parallel vs. Serial Transfer

- 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

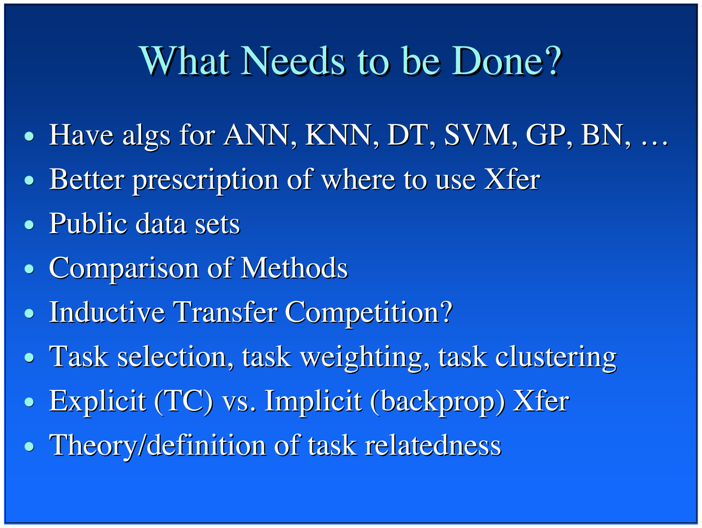


Transfer through the Ages

- 1986: Sejnowski & Rosenberg – NETtalk
- 1990: Dietterich, Hild, Bakiri – ID3 vs. NETtalk
- 1990: Suddarth, Kergiosen, & Holden – rule injection (ANNs)
- 1990: Abu-Mostafa – hints (ANNs)
- 1991: Dean Pomerleau – ALVINN output representation (ANNs)
- 1991: Lorien Pratt – speedup learning (ANNs)
- 1992: Sharkey & Sharkey – speedup learning (ANNs)
- 1992: Mark Ring – continual learning
- 1993: Rich Caruana – MTL (ANNs, KNN, DT)
- 1993: Thrun & Mitchell – EBNN
- 1994: Virginia de Sa – minimizing disagreement
- 1994: Jonathan Baxter – representation learning (and theory)
- 1994: Thrun & Mitchell – learning one more thing
- 1994: J. Schmidhuber – learning how to learn learning strategies

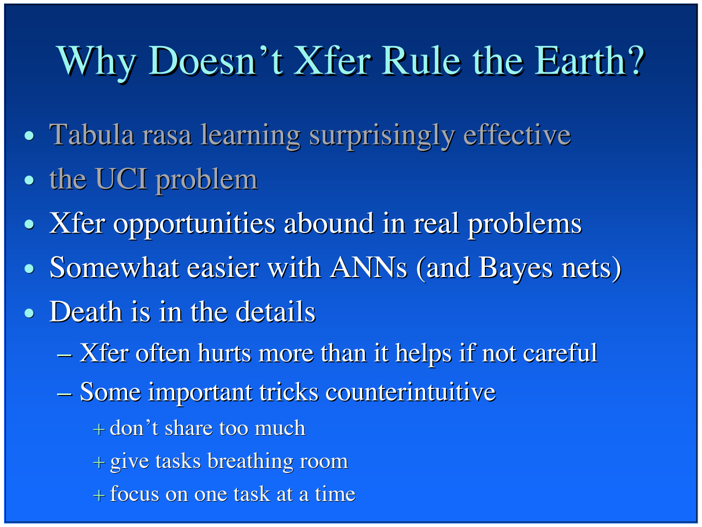


- 1994: Dietterich & Bakiri: ECOC outputs
- 1995: Breiman & Friedman – Curds & Whey
- 1995: Sebastian Thrun – LLL (learning-to-learn, lifelong-learning)
- 1996: Danny Silver – parallel transfer (ANNs)
- 1996: O'Sullivan & Thrun – task clustering (KNN)
- 1996: Caruana & de Sa – inputs better as outputs (ANNs)
- 1997: Munro & Parmanto – committee machines (ANNs)
- 1998: Blum & Mitchell – co-training
- 2002: Ben-David, Gehrke, Schuller – theoretical framework
- 2003: Bakker & Heskes – Bayesian MTL (and task clustering)
- 2004: Tony Jebara – MTL in SVMs (feature and kernel selection)
- 2004: Pontil & Micchelli – Kernels for MTL
- 2004: Lawrence & Platt – MTL in GP (info vector machine)
- 2005: Yu, Tresp, Schwaighofer – MTL in GP
- 2005: Lia & Carin – MTL for RBF Networks



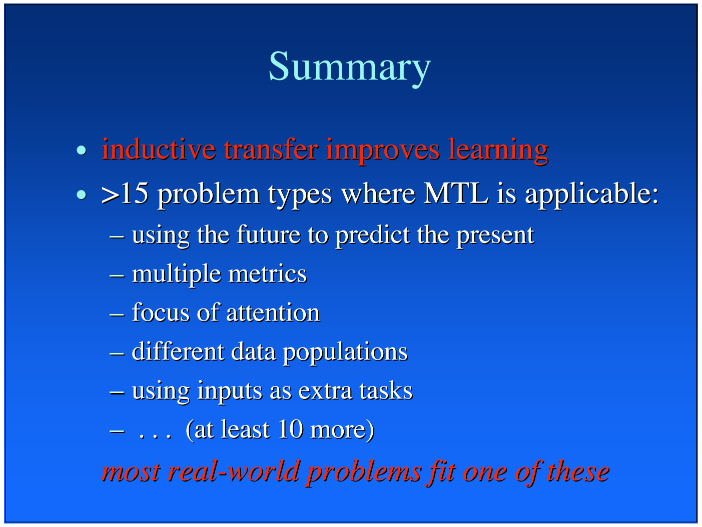
What Needs to be Done?

- Have algs for ANN, KNN, DT, SVM, GP, BN, …
- Better prescription of where to use Xfer
- Public data sets
- Comparison of Methods
- Inductive Transfer Competition?
- Task selection, task weighting, task clustering
- Explicit (TC) vs. Implicit (backprop) Xfer
- Theory/definition of task relatedness



Why Doesn't Xfer Rule the Earth?

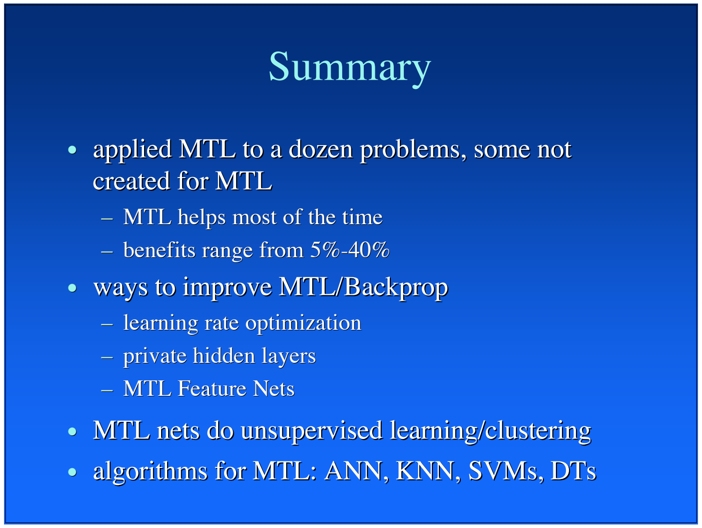
- Tabula rasa learning surprisingly effective
- the UCI problem
- Xfer opportunities abound in real problems
- Somewhat easier with ANNs (and Bayes nets)
- Death is in the details
  - Xfer often hurts more than it helps if not careful
  - Some important tricks counterintuitive
    - + don't share too much
    - + give tasks breathing room
    - + focus on one task at a time



Summary

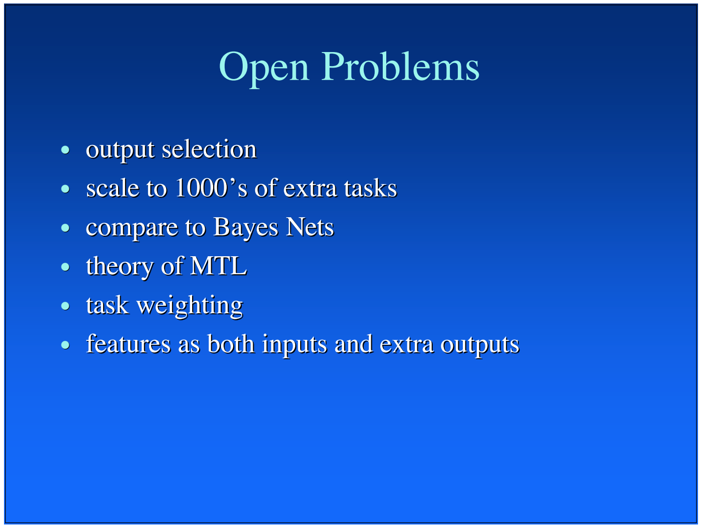
- inductive transfer improves learning
- >15 problem types where MTL is applicable:
  - using the future to predict the present
  - multiple metrics
  - focus of attention
  - different data populations
  - using inputs as extra tasks
  - . . . (at least 10 more)

most real-world problems fit one of these



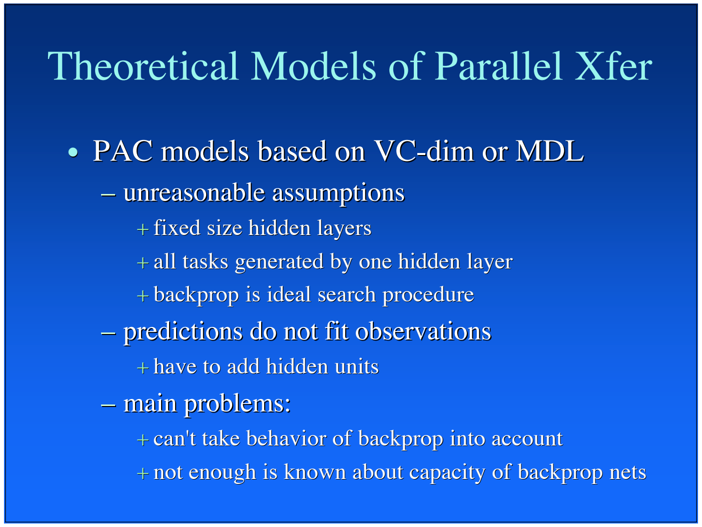
Summary

- 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 learning/clustering
- algorithms for MTL: ANN, KNN, SVMs, DTs



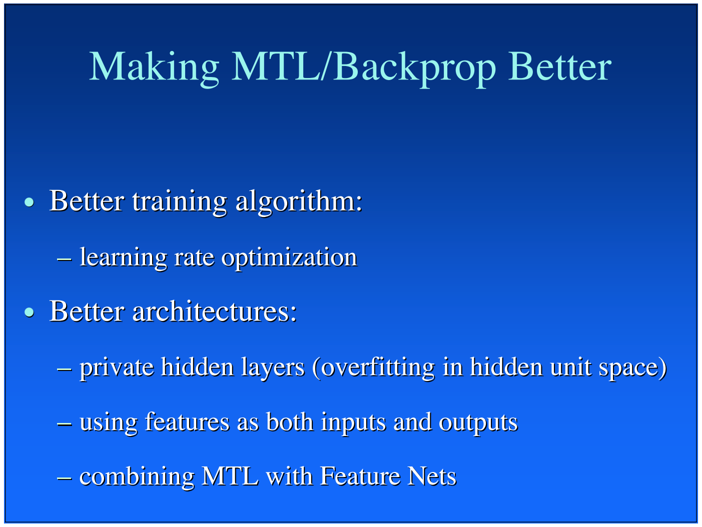
Open Problems

- output selection
- scale to 1000's of extra tasks
- compare to Bayes Nets
- theory of MTL
- task weighting
- features as both inputs and extra outputs



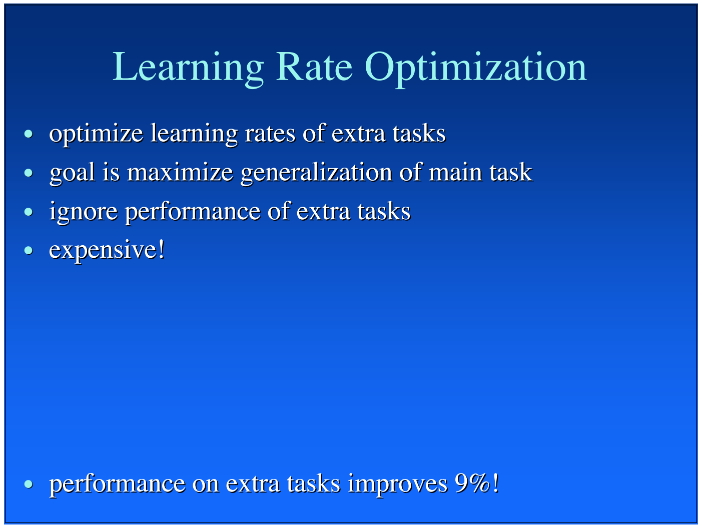
Theoretical Models of Parallel Xfer

- 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



Making MTL/Backprop Better

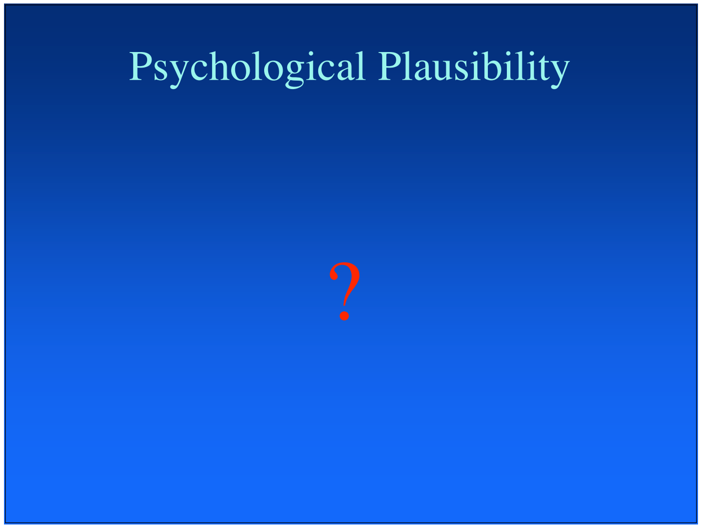
- Better training algorithm:
  - – learning rate optimization
- Better architectures:
  - – private hidden layers (overfitting in hidden unit space)
  - – using features as both inputs and outputs
  - – combining MTL with Feature Nets



Learning Rate Optimization

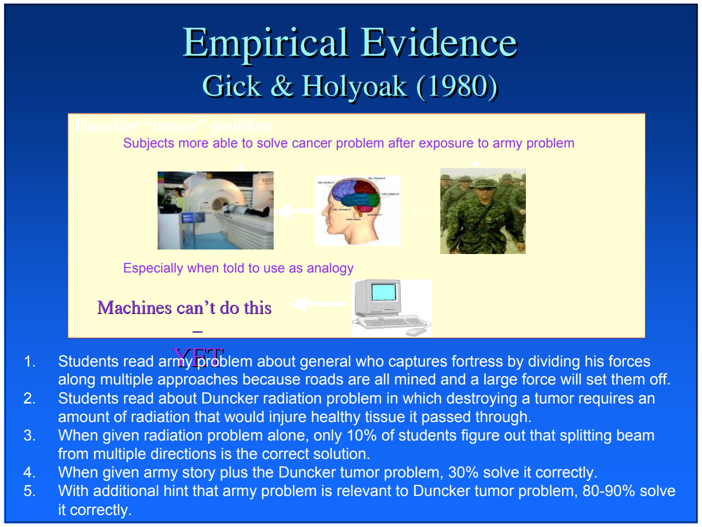
- optimize learning rates of extra tasks
- goal is maximize generalization of main task
- ignore performance of extra tasks
- expensive!

- performance on extra tasks improves 9%!



Psychological Plausibility

?



Empirical Evidence

Gick & Holyoak (1980)

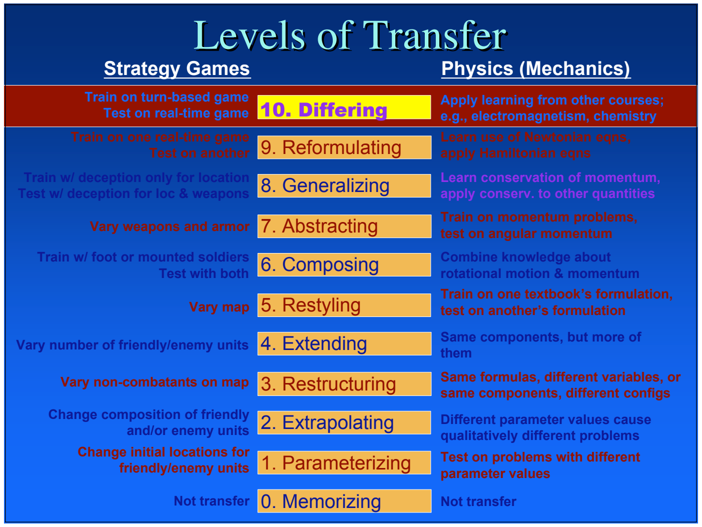
Duncker "tumor" problem

Subjects more able to solve cancer problem after exposure to army problem

Especially when told to use as analogy

Machines can't do this – YET

1. Students read army problem about general who captures fortress by dividing his forces along multiple approaches because roads are all mined and a large force will set them off.
2. Students read about Duncker radiation problem in which destroying a tumor requires an amount of radiation that would injure healthy tissue it passed through.
3. When given radiation problem alone, only 10% of students figure out that splitting beam from multiple directions is the correct solution.
4. When given army story plus the Duncker tumor problem, 30% solve it correctly.
5. With additional hint that army problem is relevant to Duncker tumor problem, 80-90% solve it correctly.



Levels of Transfer

Strategy Games		Physics (Mechanics)
Train on turn-based game Test on real-time game	10. Differing	Apply learning from other courses; e.g., electromagnetism, chemistry
Train on one real-time game Test on another	9. Reformulating	Learn use of Newtonian eqns, apply Hamiltonian eqns
Train w/ deception only for location Test w/ deception for loc & weapons	8. Generalizing	Learn conservation of momentum, apply conserv. to other quantities
Vary weapons and armor	7. Abstracting	Train on momentum problems, test on angular momentum
Train w/ foot or mounted soldiers Test with both	6. Composing	Combine knowledge about rotational motion & momentum
Vary map	5. Restyling	Train on one textbook's formulation, test on another's formulation
Vary number of friendly/enemy units	4. Extending	Same components, but more of them
Vary non-combatants on map	3. Restructuring	Same formulas, different variables, or same components, different configs
Change composition of friendly and/or enemy units	2. Extrapolating	Different parameter values cause qualitatively different problems
Change initial locations for friendly/enemy units	1. Parameterizing	Test on problems with different parameter values
Not transfer	0. Memorizing	Not transfer