Multi-Task Learning as a Bargaining Game

Aviv Navon*[1] Aviv Shamsian*[1] Idan Achituve[1] Haggai Maron[1,2]
Kenji Kawaguchi[3] Gal Chechik[1,2] Ethan Fetaya[1]

[1]Bar-Ilan University [2]NVIDIA Research [3]National University of Singapore

Overview

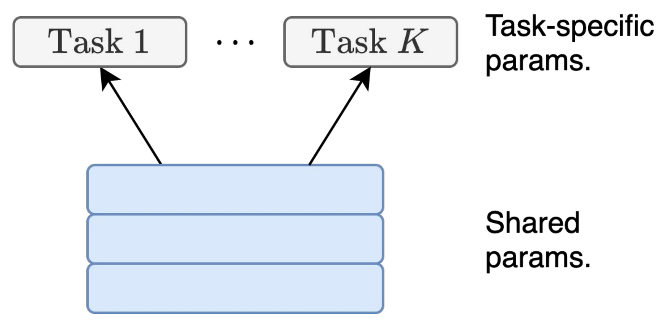
Training multiple tasks jointly can reduce computation costs and improve data efficiency, but it has a major challenge: gradients tend to conflict in direction and differ in magnitude.
In *multi-task learning* (MTL) it is not clear how to combine the gradients into a joint update direction.

We propose *Nash-MTL*, a principled MTL approach, that views the gradient aggregation step as a bargaining game.

Multi-task Learning

In MTL, a joint model is trained to simultaneously make predictions for several tasks. Generally:

- All tasks share an encoder (feature extractor).
- Each task has a task-specific head.



Compared with single-task (STL) models, MTL can potentially:

- Reduce computation costs at inference.
- Improve generalization and data efficiency.

A Common Approach to MTL

Most MTL optimization algorithms follow:

1. *Differentiate*: compute per-task gradients $g_i, i = 1, ..., K$.
2. *Aggregate*: combine gradients into a joint direction Δ using aggregation alg. $\mathcal{A}$.
3. *Update* the parameters according to Δ.

Our novel MTL algorithm views the aggregation step as a Bargaining game.

A Bargaining Game

- K players, each equipped with its own utility function u_i.
- Players must find a point they all agree upon or default to the disagreement point.
- Nash proposed an *axiomatic* approach and proved a unique solution exists with desired properties like *Pareto optimality* and *symmetry*.
- This unique solution is called the ***Nash bargaining solution***.

Algorithm 1 Nash-MTL

Input: $\theta^{(0)}$ – initial parameter vector, $\{\ell_i\}_{i=1}^K$ – differentiable loss functions, η – learning rate
for $t = 1, ..., T$ **do**
  Compute task gradients $g_i^{(t)} = \nabla_{\theta^{(t-1)}} \ell_i$
  Set $G^{(t)}$ the matrix with columns $g_i^{(t)}$
  Solve for α: $(G^{(t)})^\top G^{(t)} \alpha = 1/\alpha$ to obtain $\alpha^{(t)}$
  Update the parameters $\theta^{(t)} = \theta^{(t)} - \eta G^{(t)} \alpha^{(t)}$
end for
Return: $\theta^{(T)}$

Nash-MTL

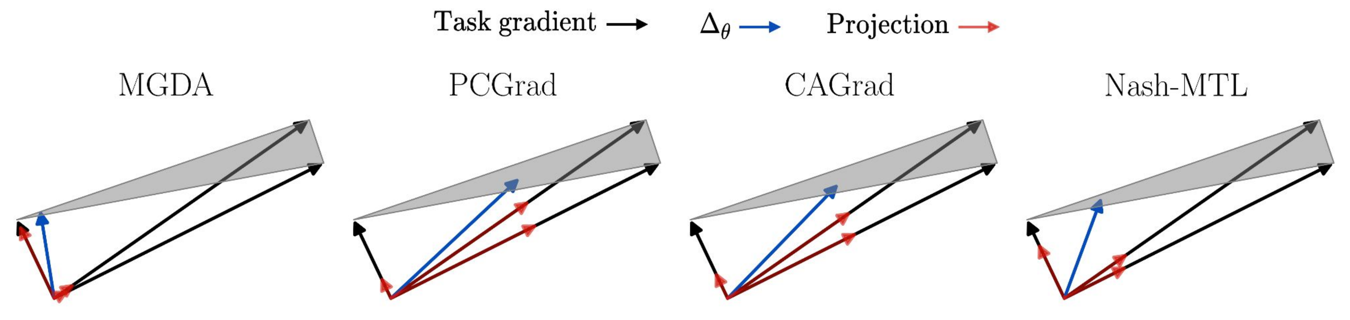
Our approach, *Nash-MTL*, uses the Nash bargaining solution as the update direction in MTL.

- Given an MTL problem with parameters θ.
- Search for update $\Delta\theta$ direction in an ϵ-ball around the origin.
- Define the utility for task i as a directional derivative $u_i(\Delta\theta) = \Delta\theta^T g_i$.
- Denote G the matrix whose columns are g_i.

Claim: *The Nash bargaining solution for MTL is given by* $\Delta\theta = \sum_i \alpha_i g_i$ *s.t.* $G^T G \alpha = 1/\alpha$ *where* $1/\alpha$ *is taken element-wise.*

Theoretical analysis: We prove that the sequence generated by our method converges to a Pareto optimal (stationary) point in the (non-convex) convex case.

Illustrative Visualization of the Update Direction

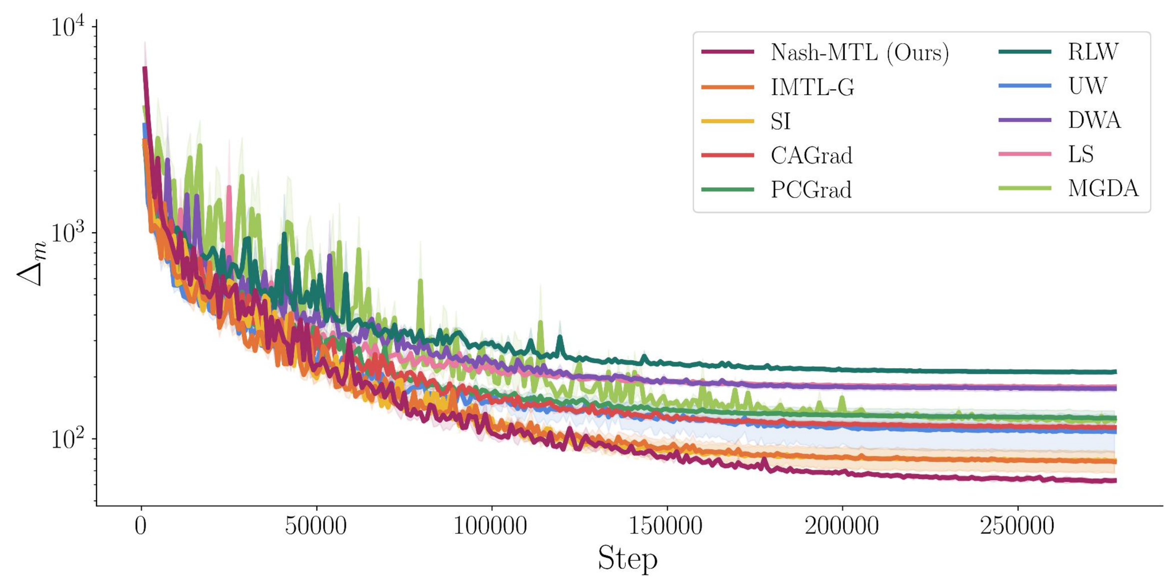


- Update direction obtained by various methods on three gradients.
- Nash-MTL produce an update direction, colored in blue, with the most balanced per-tasks projections, marked in red.

Multi-task Regression

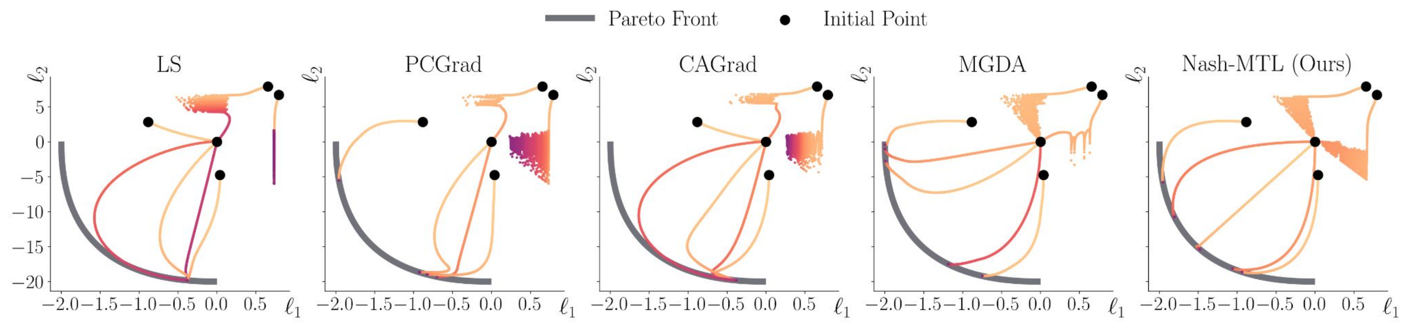
We evaluate *Nash-MTL* on predicting 11 properties of molecules from the QM9 dataset.

- QM9 poses a significant challenge for MTL methods since the number of tasks is large and because the loss scales vary significantly.
- *Nash-MTL* achieves the best performance.



Paper Code Project page

Illustrative Example



We consider a problem with two losses of different scales and plot the optimization trajectory in objective space.

- Nash-MTL can find well balanced optimal solutions.

Scene Understanding

We evaluate Nash-MTL on NYUv2, a scene understanding problem with three tasks.

- *Nash-MTL* achieves the best overall results.

	Segmentation		Depth		Surface Normal						
	mIoU ↑	Pix Acc ↑	Abs Err ↓	Rel Err ↓	Angle Distance ↓		Within t° ↑			**MR** ↓	**Δm%** ↓
					Mean	Median	11.25	22.5	30		
STL	38.30	63.76	0.6754	0.2780	25.01	19.21	30.14	57.20	69.15		
LS	39.29	65.33	0.5493	0.2263	28.15	23.96	22.09	47.50	61.08	8.11	5.59
SI	38.45	64.27	0.5354	0.2201	27.60	23.37	22.53	48.57	62.32	7.11	4.39
RLW	37.17	63.77	0.5759	0.2410	28.27	24.18	22.26	47.05	60.62	10.11	7.78
DWA	39.11	65.31	0.5510	0.2285	27.61	23.18	24.17	50.18	62.39	6.88	3.57
UW	36.87	63.17	0.5446	0.2260	27.04	22.61	23.54	49.05	63.65	6.44	4.05
MGDA	30.47	59.90	0.6070	0.2555	**24.88**	**19.45**	**29.18**	**56.88**	**69.36**	5.44	1.38
PCGrad	38.06	64.64	0.5550	0.2325	27.41	22.80	23.86	49.83	63.14	6.88	3.97
GradDrop	39.39	65.12	0.5455	0.2279	27.48	22.96	23.38	49.44	62.87	6.44	3.58
CAGrad	39.79	65.49	0.5486	0.2250	26.31	21.58	25.61	52.36	65.58	3.77	0.20
IMTL-G	39.35	65.60	0.5426	0.2256	26.02	21.19	26.2	53.13	66.24	3.11	−0.76
Nash-MTL	**40.13**	**65.93**	**0.5261**	**0.2171**	25.26	20.08	28.4	55.47	68.15	**1.55**	**−4.04**

Multi-task RL

- MT10 environment with ten tasks.
- *Nash-MTL* achieves the best performance by a large margin.
- It is the only multitask method to outperform STL.

	Success ± SEM
STL SAC	0.90 ± 0.032
MTL SAC	0.49 ± 0.073
MTL SAC + TE	0.54 ± 0.047
MH SAC	0.61 ± 0.036
SM	0.73 ± 0.043
CARE	0.84 ± 0.051
PCGrad	0.72 ± 0.022
CAGrad	0.83 ± 0.045
Nash-MTL	**0.91 ± 0.031**