



復旦大學



# Learning Sparse Sharing Architectures for Multiple Tasks

Tianxiang Sun

[txsun19@fudan.edu.cn](mailto:txsun19@fudan.edu.cn)

# Outline



Multi-Task Learning: Sharing Mechanisms

Sparse Sharing Mechanism

Approach: Learning Sparse Sharing Architectures

Experiments

Analysis and Discussions

Conclusion

# Outline



Multi-Task Learning: Sharing Mechanisms

Sparse Sharing Mechanism

Approach: Learning Sparse Sharing Architectures

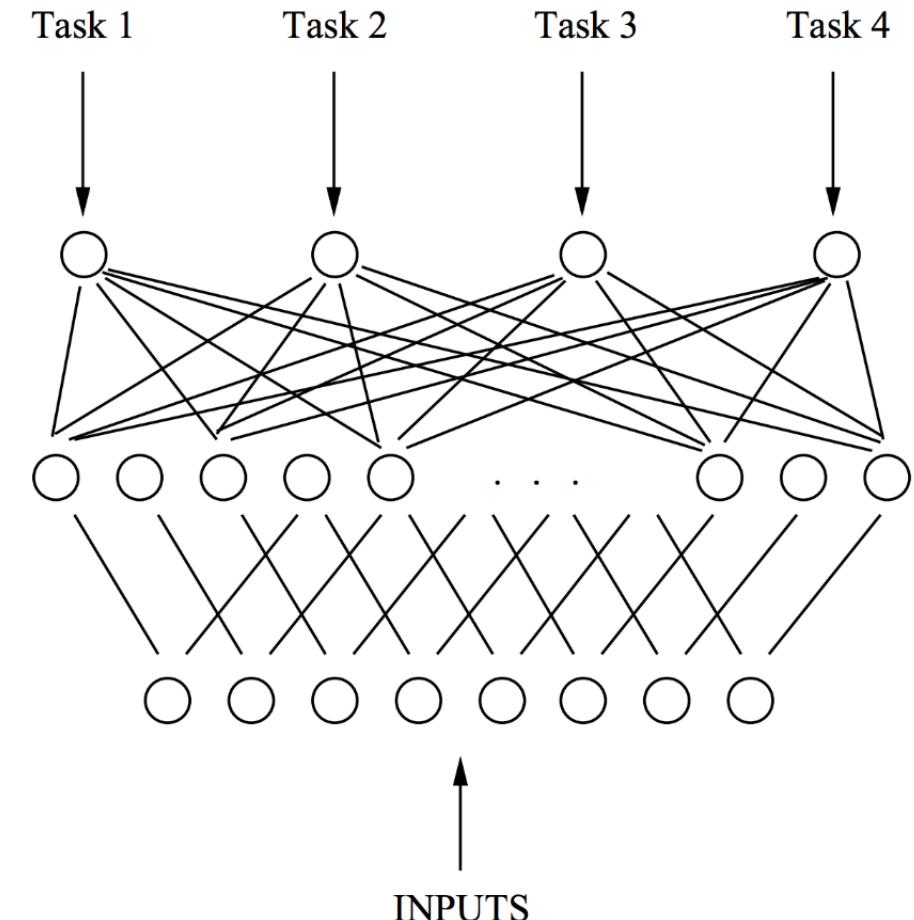
Experiments

Analysis and Discussions

Conclusion

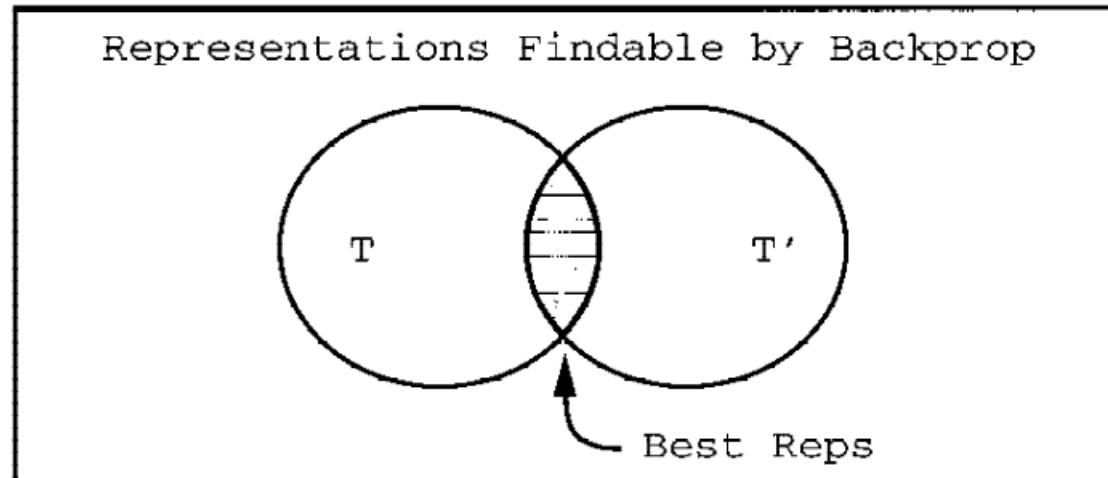
# What is Multi-Task Learning(MTL)?

“Multi-task Learning is an approach to inductive transfer that improves **generalization** by using the domain information contained in the training signals of **related tasks** as an **inductive bias**.”



# How does MTL work?

- Representation Bias (Inductive Bias)



# Formulation

- $T$  tasks:  $\mathcal{D}_t = \{x_n^t, y_n^t\}_{n=1}^{N_t}$
- Shared layers  $\mathcal{E}$  parameterized by  $\theta_{\mathcal{E}} = \{\theta_{\mathcal{E},1}, \dots, \theta_{\mathcal{E},L}\}$
- Task-specific layers  $\mathcal{F}^t$  parameterized by  $\theta_{\mathcal{F}}^t$
- Parameters:  $\theta = (\theta_{\mathcal{E}}, \theta_{\mathcal{F}}^1, \dots, \theta_{\mathcal{F}}^T)$
- Objective:  $\mathcal{L}(\theta) = \sum_{t=1}^T \lambda_t \sum_{n=1}^{N_t} \mathcal{L}_t(\hat{y}_n^t, y_n^t)$

# Multi-Task Sharing Mechanisms



MTL is typically done with *parameter sharing*:

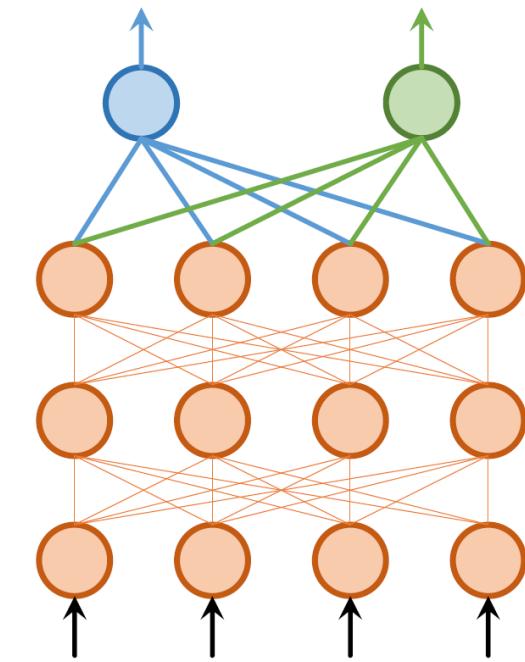
- Hard Sharing ([Collobert and Weston 2008](#); [Subramanian et al. 2018](#); [Liu et al. 2019](#))
- Soft Sharing ([Misra et al. 2016](#); [Ruder et al. 2019](#))
- Hierarchical Sharing ([Søgaard and Goldberg 2016](#); [Hashimoto et al. 2017](#))
- ...

# Hard Sharing



- Stack the task-specific layers on the top of the shared layers
- Inference:  $\hat{y}_n^t = \mathcal{F}^t(\mathcal{E}(x_n^t; \theta_{\mathcal{E}}); \theta_{\mathcal{F}}^t)$
- Advantages:
  1. simple to implement
  2. parameter efficient
- Disadvantages:

Struggle with loosely related/unrelated tasks  
**(Negative Transfer)**



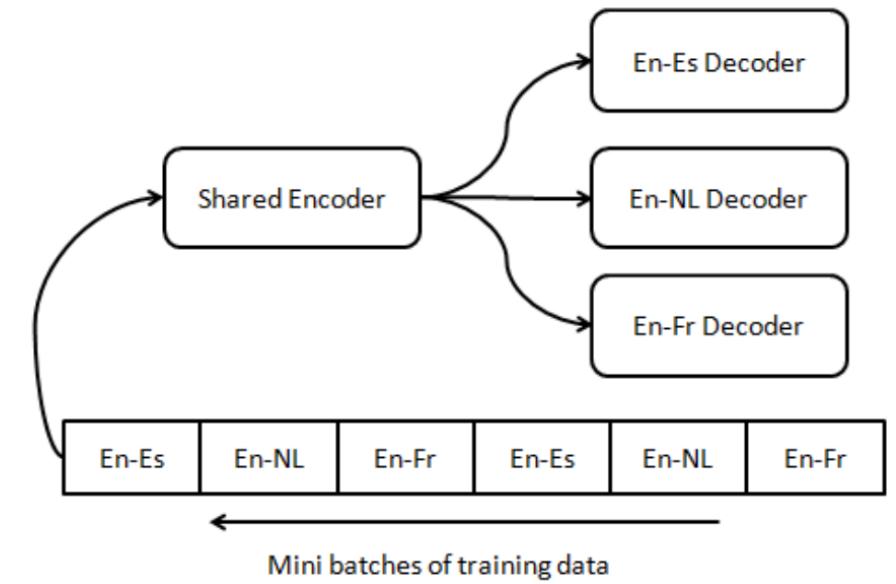
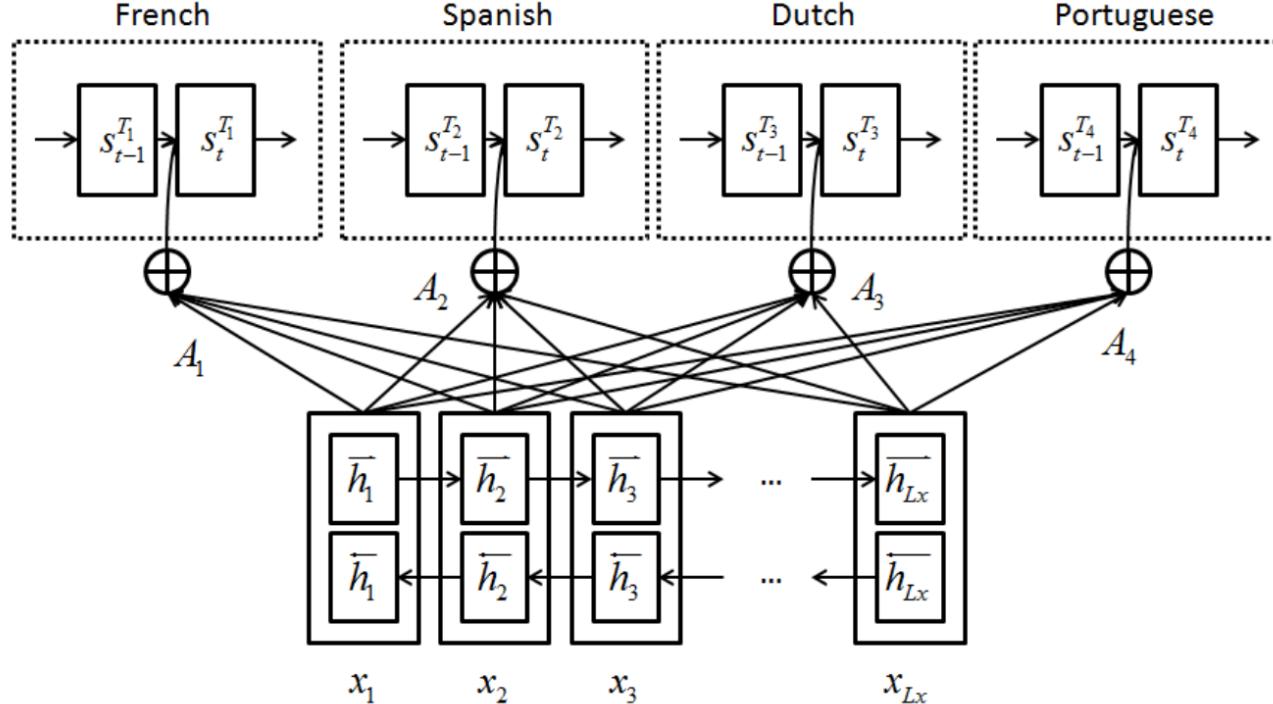
# Multi-Task Learning for Multiple Language Translation

ACL 2015

Daxiang Dong, Hua Wu, Wei He, Dianhai Yu and Haifeng Wang

Baidu Inc, Beijing, China

{dongdaxiang, wu\_hua, hewei06, yudianhai, wanghaifeng}@baidu.com



# Recurrent Neural Network for Text Classification with Multi-Task Learning

IJCAI 2016

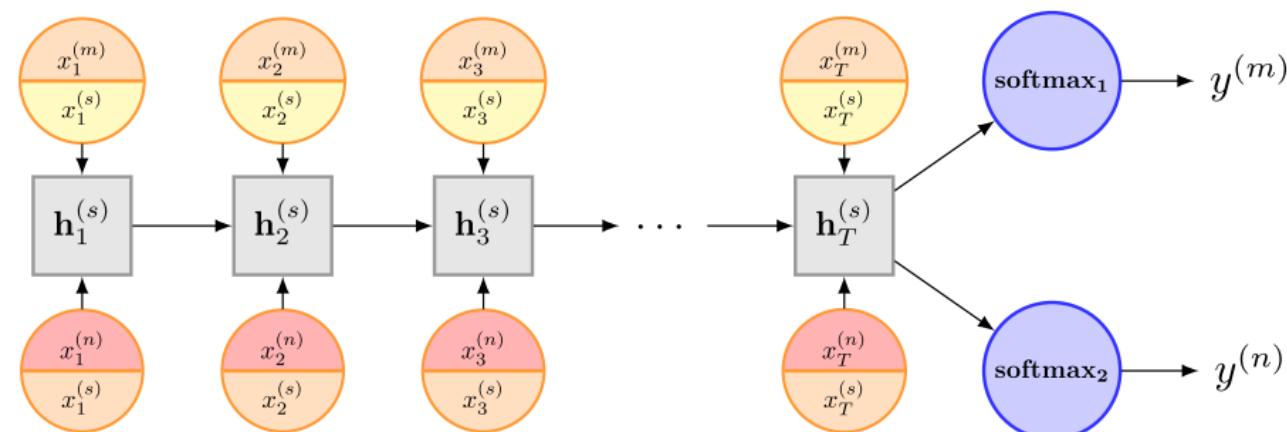
Pengfei Liu Xipeng Qiu\* Xuanjing Huang

Shanghai Key Laboratory of Intelligent Information Processing, Fudan University

School of Computer Science, Fudan University

825 Zhangheng Road, Shanghai, China

{pfliu14,xpqiu,xjhuang}@fudan.edu.cn

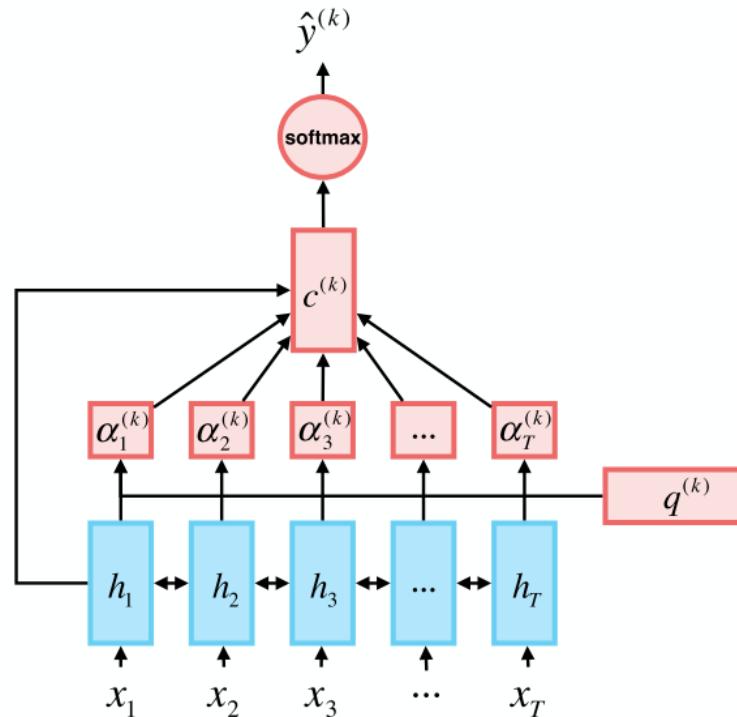


# Same Representation, Different Attentions: Shareable Sentence Representation Learning from Multiple Tasks

IJCAI 2018

Renjie Zheng, Junkun Chen, Xipeng Qiu\*

Shanghai Key Laboratory of Intelligent Information Processing, Fudan University  
School of Computer Science, Fudan University  
825 Zhangheng Road, Shanghai, China

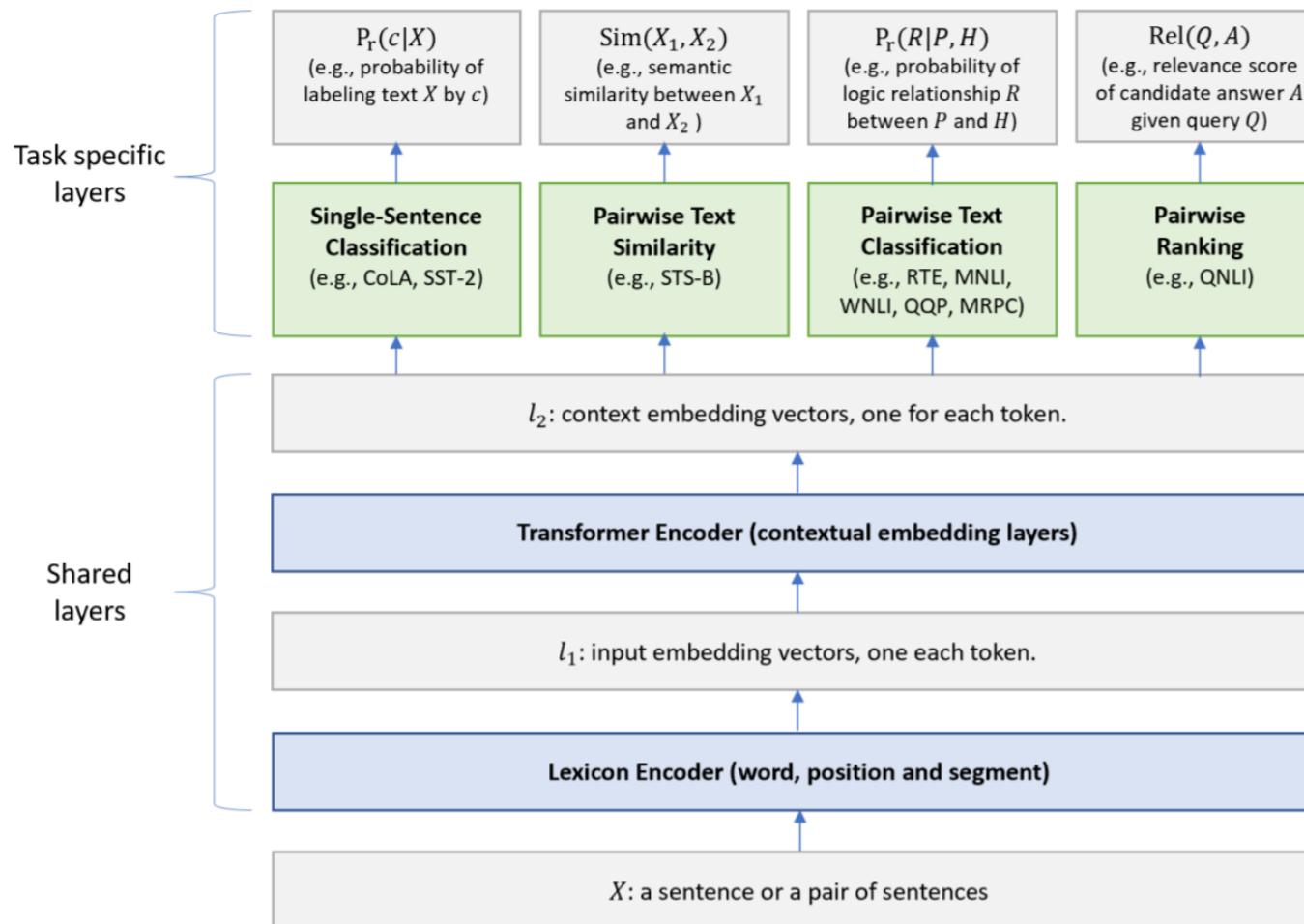


Xiaodong Liu<sup>\*1</sup>, Pengcheng He<sup>\*2</sup>, Weizhu Chen<sup>2</sup>, Jianfeng Gao<sup>1</sup>

<sup>1</sup> Microsoft Research

<sup>2</sup> Microsoft Dynamics 365 AI

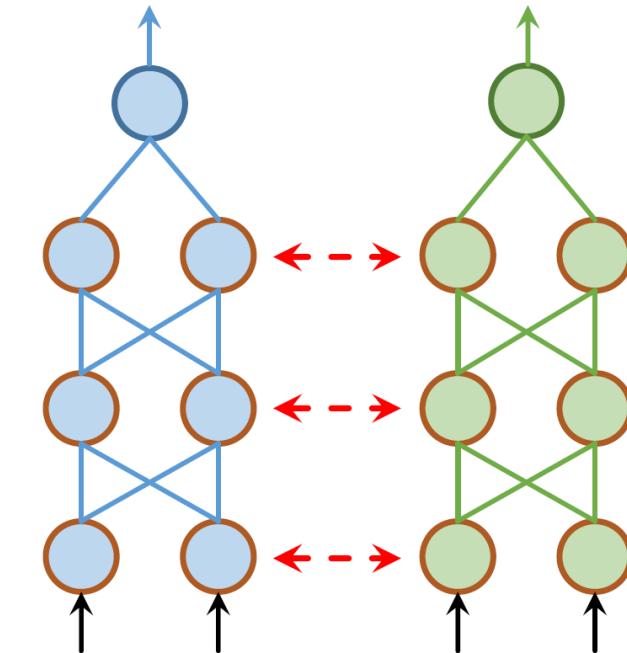
{xiaodl, penhe, wzchen, jfgao}@microsoft.com



# Soft Sharing



- Each task has separate model and parameters, but each model can access the information inside other models
- Advantages:  
Makes no assumptions about task relatedness
- Disadvantages:  
Not parameter-efficient



# Recurrent Neural Network for Text Classification with Multi-Task Learning

IJCAI 2016

Pengfei Liu Xipeng Qiu\* Xuanjing Huang

Shanghai Key Laboratory of Intelligent Information Processing, Fudan University

School of Computer Science, Fudan University

825 Zhangheng Road, Shanghai, China

{pfliu14,xpqiu,xjhuang}@fudan.edu.cn

## ➤ Single-Task LSTM

$$\mathbf{i}_t = \sigma(\mathbf{W}_i \mathbf{x}_t + \mathbf{U}_i \mathbf{h}_{t-1} + \mathbf{V}_i \mathbf{c}_{t-1}),$$

$$\mathbf{f}_t = \sigma(\mathbf{W}_f \mathbf{x}_t + \mathbf{U}_f \mathbf{h}_{t-1} + \mathbf{V}_f \mathbf{c}_{t-1}),$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_o \mathbf{x}_t + \mathbf{U}_o \mathbf{h}_{t-1} + \mathbf{V}_o \mathbf{c}_t),$$

$$\tilde{\mathbf{c}}_t = \tanh(\mathbf{W}_c \mathbf{x}_t + \mathbf{U}_c \mathbf{h}_{t-1}),$$

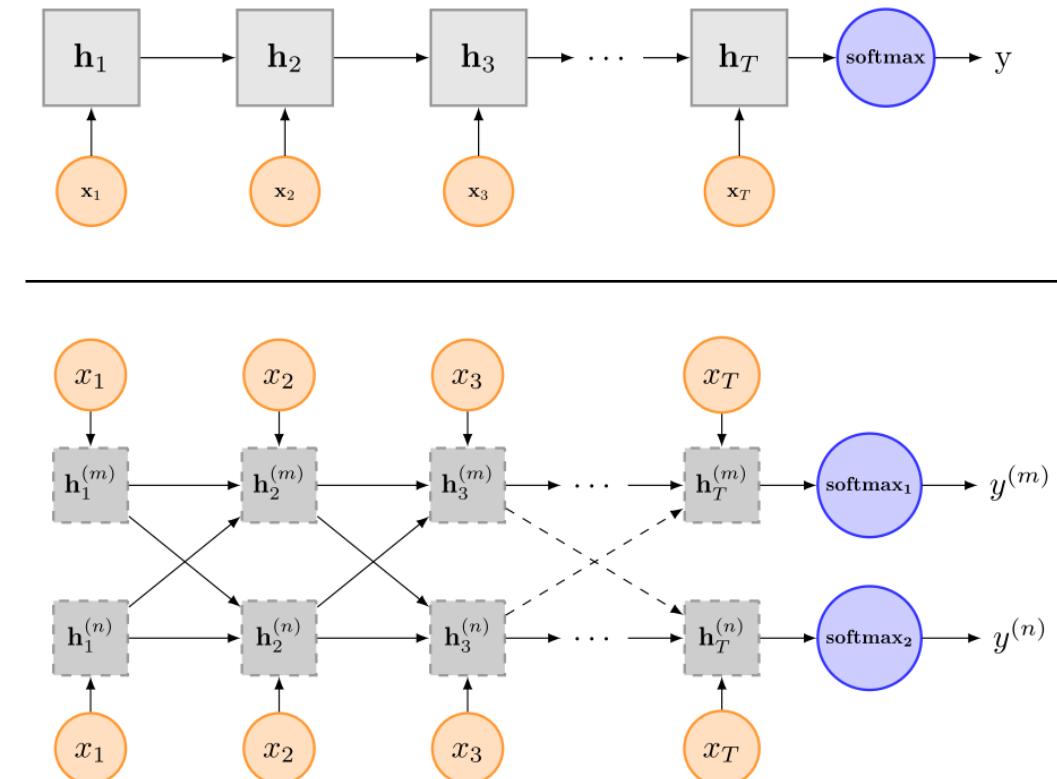
$$\mathbf{c}_t = \mathbf{f}_t^i \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t,$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t),$$

## ➤ Add signal flow across tasks

$$\tilde{\mathbf{c}}_t^{(m)} = \tanh \left( \mathbf{W}_c^{(m)} \mathbf{x}_t + \sum_{i \in \{m, n\}} \mathbf{g}^{(i \rightarrow m)} U_c^{(i \rightarrow m)} \mathbf{h}_{t-1}^{(i)} \right)$$

$$\mathbf{g}^{(i \rightarrow m)} = \sigma(\mathbf{W}_g^{(m)} \mathbf{x}_t + \mathbf{U}_g^{(i)} \mathbf{h}_{t-1}^{(i)})$$



# Cross-stitch Networks for Multi-task Learning

CVPR 2016

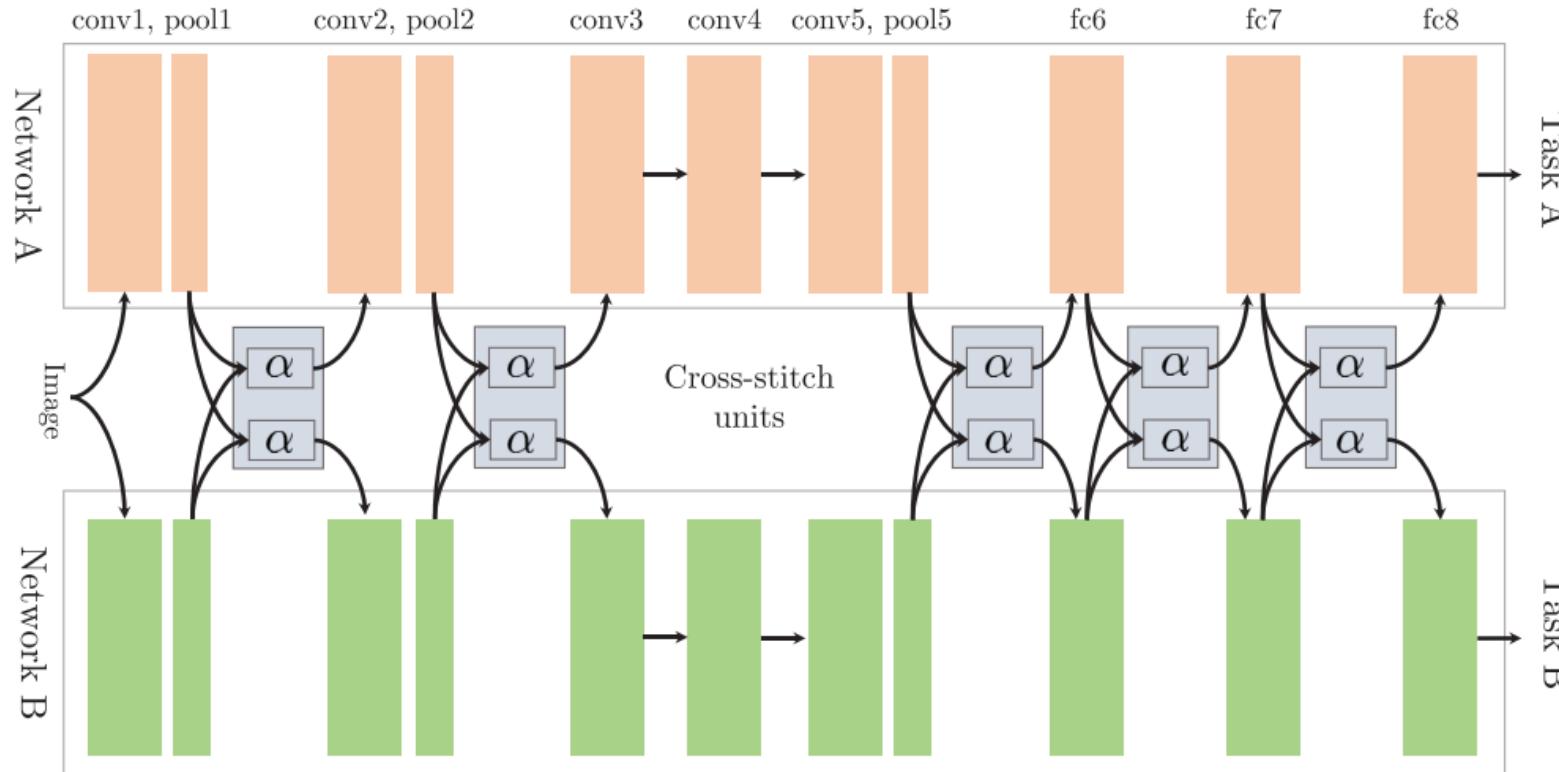
Ishan Misra\*

Abhinav Shrivastava\*

Abhinav Gupta

Martial Hebert

The Robotics Institute, Carnegie Mellon University



$$\begin{bmatrix} \tilde{x}_A^{ij} \\ \tilde{x}_B^{ij} \end{bmatrix} = \begin{bmatrix} \alpha_{AA} & \alpha_{AB} \\ \alpha_{BA} & \alpha_{BB} \end{bmatrix} \begin{bmatrix} x_A^{ij} \\ x_B^{ij} \end{bmatrix}$$

# Latent Multi-task Architecture Learning

AAAI 2019

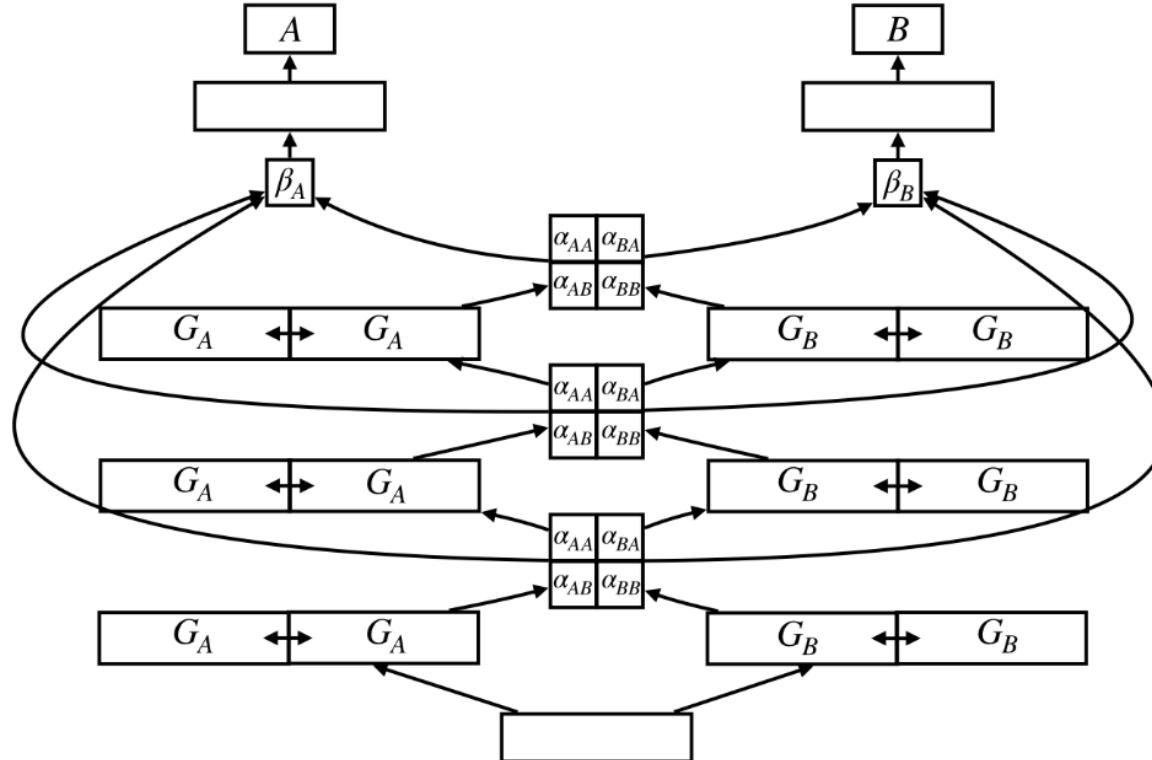
**Sebastian Ruder<sup>12</sup>, Joachim Bingel<sup>3</sup>, Isabelle Augenstein<sup>3</sup>, Anders Søgaard<sup>3</sup>**

<sup>1</sup>Insight Research Centre, National University of Ireland, Galway

<sup>2</sup>Aylien Ltd., Dublin, Ireland

<sup>3</sup>Department of Computer Science, University of Copenhagen, Denmark

sebastian@ruder.io, {bingel|augenstein|soegaard}@di.ku.dk



$$\begin{bmatrix} \tilde{h}_{A_1,k} \\ \vdots \\ \tilde{h}_{B_2,k} \end{bmatrix} = \begin{bmatrix} \alpha_{A_1A_1} & \dots & \alpha_{B_2A_1} \\ \vdots & \ddots & \vdots \\ \alpha_{A_1B_2} & \dots & \alpha_{B_2B_2} \end{bmatrix} [h_{A_1,k}^\top, \dots, h_{B_2,k}^\top]$$

---

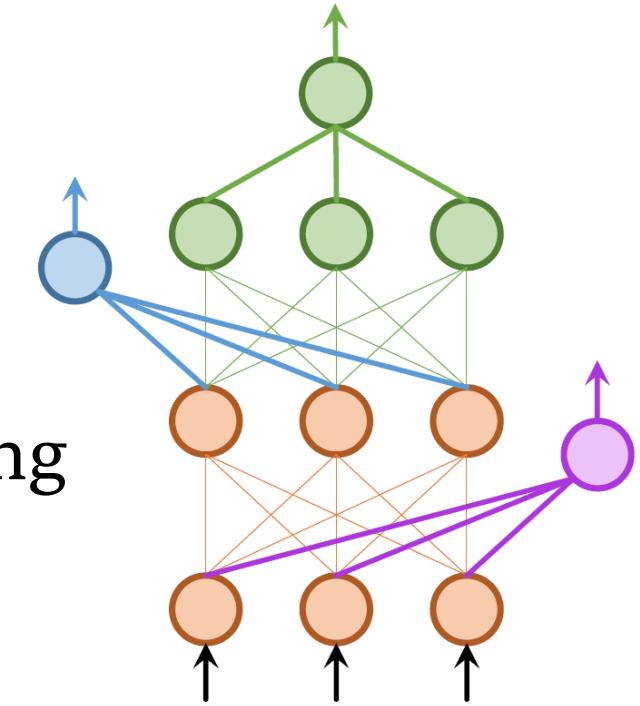
$$\tilde{h}_A^\top = \begin{bmatrix} \beta_{A,1} \\ \vdots \\ \beta_{A,k} \end{bmatrix}^\top [h_{A,1}^\top, \dots, h_{A,k}^\top]$$

# Hierarchical Sharing



- Put different task supervisions at different layers
- Inference:  $\hat{y}_n^t = \mathcal{F}^t(\mathcal{E}(x_n^t; \theta_{\mathcal{E}(1:l)}); \theta_{\mathcal{F}}^t)$
- Advantages:
  1. more flexible than hard sharing
  2. more parameter-efficient than soft sharing
- Disadvantages:

Hard to design an effective hierarchy



**Anders Søgaard**  
University of Copenhagen  
soegaard@hum.ku.dk

**Yoav Goldberg**  
Bar-Ilan University  
yoav.goldberg@gmail.com

$$\begin{aligned} pos\_tag(w_{1:n}, i) &= f_{pos}(v_i^{\ell(pos)}) \\ chunk\_tag(w_{1:n}, i) &= f_{chunk}(v_i^{\ell(chunk)}) \\ ccg\_tag(w_{1:n}, i) &= f_{ccg}(v_i^{\ell(ccg)}) \\ v_i^\ell &= BIRNN^\ell(x_{1:n}, i) \\ x_{1:n} &= E(w_1), E(w_2), \dots, E(w_n) \end{aligned}$$

# A Joint Many-Task Model: Growing a Neural Network for Multiple NLP Tasks

EMNLP 2017

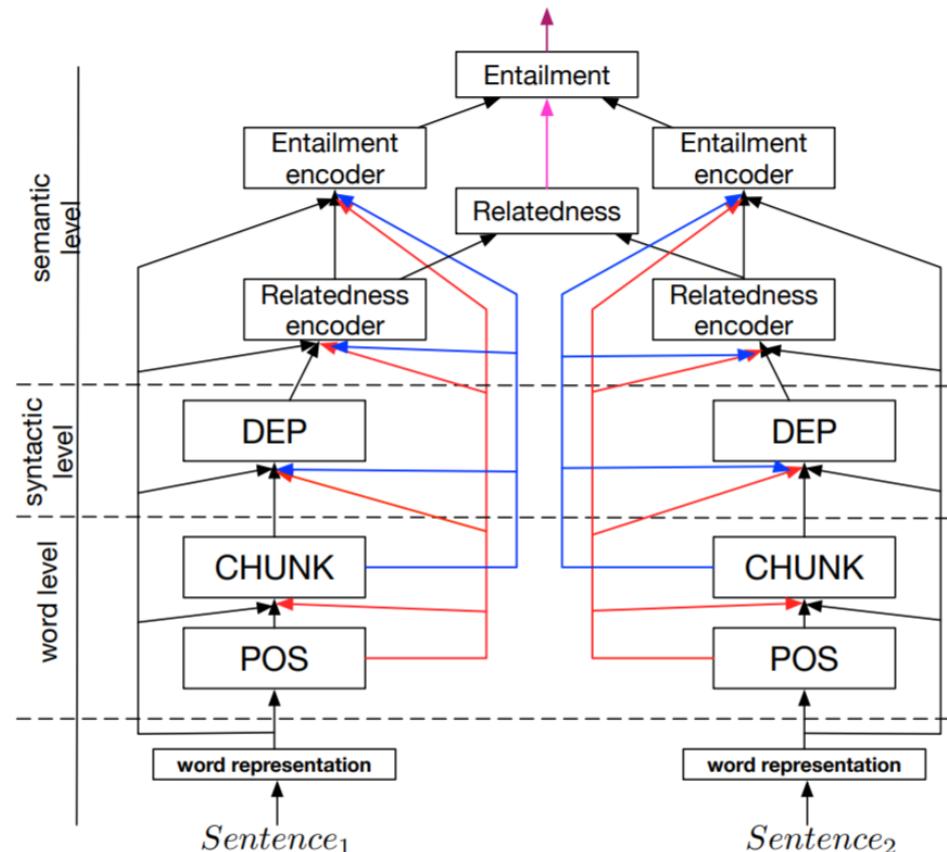
Kazuma Hashimoto\*, Caiming Xiong<sup>†</sup>, Yoshimasa Tsuruoka, and Richard Socher

The University of Tokyo

{hassy, tsuruoka}@logos.t.u-tokyo.ac.jp

Salesforce Research

{cxiong, rsocher}@salesforce.com



# A Hierarchical Multi-task Approach for Learning Embeddings from Semantic Tasks

AAAI 2019

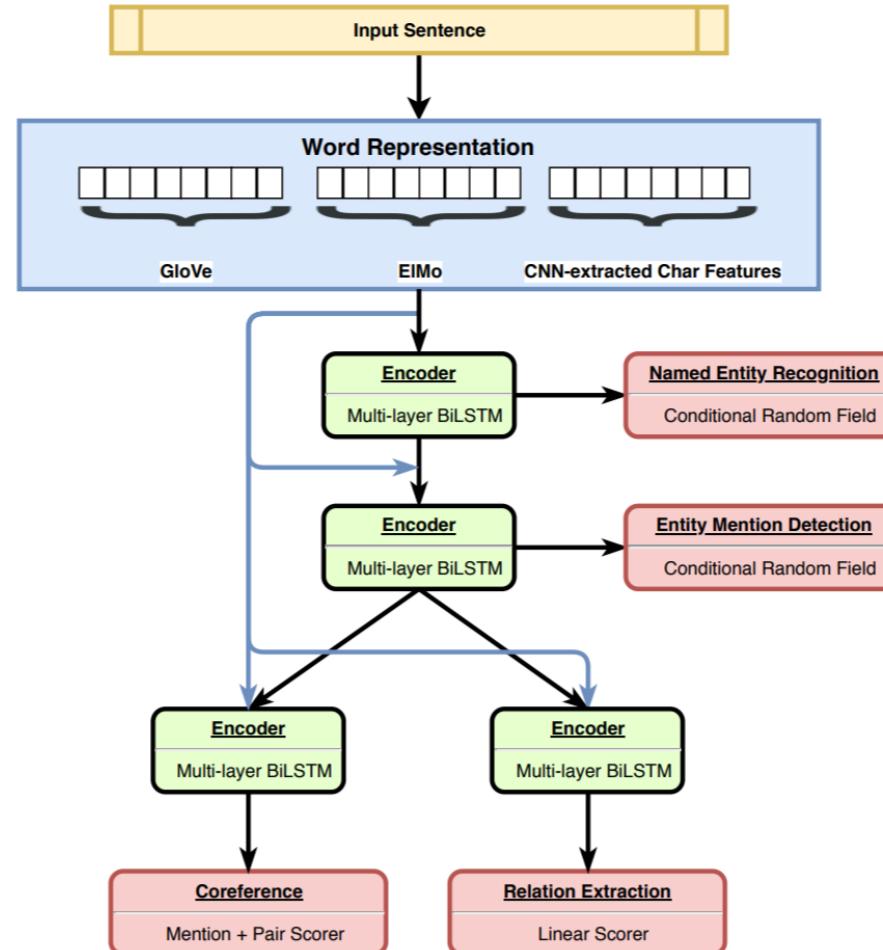
**Victor Sanh<sup>1</sup>, Thomas Wolf<sup>1</sup>, Sebastian Ruder<sup>2,3</sup>**

<sup>1</sup>Hugging Face, 20 Jay Street, Brooklyn, New York, United States

<sup>2</sup>Insight Research Centre, National University of Ireland, Galway, Ireland

<sup>3</sup>Aylien Ltd., 2 Harmony Court, Harmony Row, Dublin, Ireland

{victor, thomas}@huggingface.co, sebastian@ruder.io



# Limitations of Existing Sharing Mechanisms



- **Hard sharing:** Struggle with loosely related tasks
- **Hierarchical sharing:** Dependent on manually design
- **Soft sharing:** Parameter-inefficient



Does there exist a multi-task sharing mechanism:

1. It is compatible with a wide range of tasks, regardless of whether the tasks are related or not.
2. It does not depend on manually designing the sharing structure based on characteristic of tasks.
3. It is parameter efficient.

# Outline



Multi-Task Learning: Sharing Mechanisms

**Sparse Sharing Mechanism**

Approach: Learning Sparse Sharing Architectures

Experiments

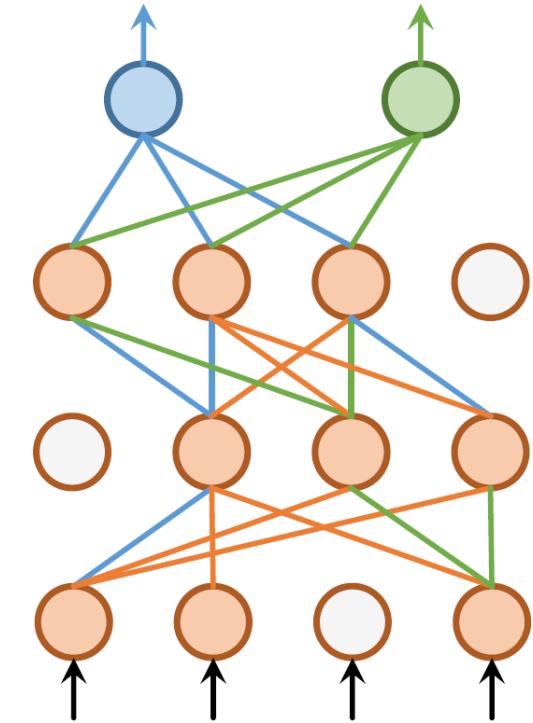
Analysis and Discussions

Conclusion

# Sparse Sharing Mechanism



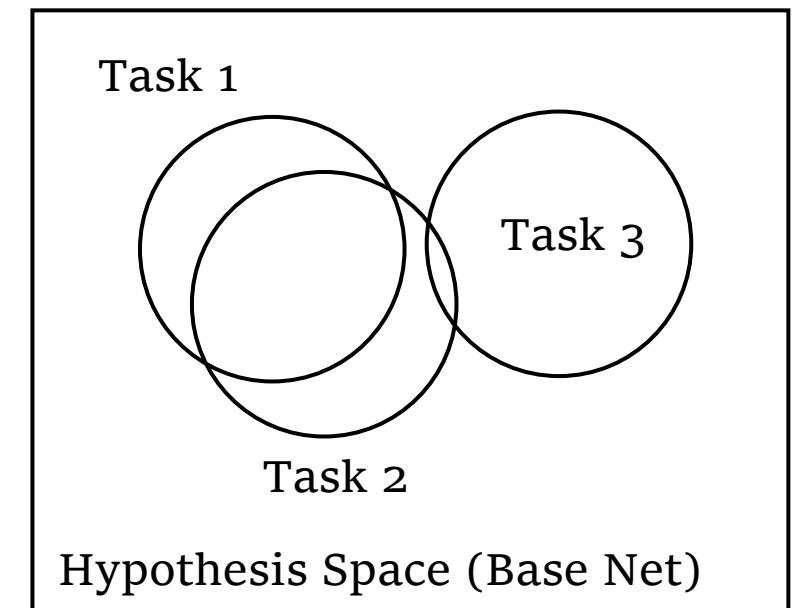
- Assign each task a subnet
- Base Network:  $\mathcal{E}$
- Subnet:  $\mathcal{E}^t(x) = \mathcal{E}(x; M_t \odot \theta_{\mathcal{E}})$
- Hard sharing  $\rightarrow M_t = \mathbf{1}$
- Hierarchical sharing  $\rightarrow$   
 $\theta_{\mathcal{E}} = \{\theta_{\mathcal{E},1}, \theta_{\mathcal{E},2}\}$     $M_1 = \{1, 0\}$     $M_2 = \{1, 1\}$



# Views of Sparse Sharing



- Over-parameterized base net → Large hypothesis space
- Subnet → Hypothesis subspace
- Inductive bias → Subnet structure
- Parameter overlap → Task relatedness
- Biologically intuitive:
  1. Sparse topology ([Pessoa 2014](#))
  2. Different subnets for different tasks ([MacLeod 1991](#))



# Outline



Multi-Task Learning: Sharing Mechanisms

Sparse Sharing Mechanism

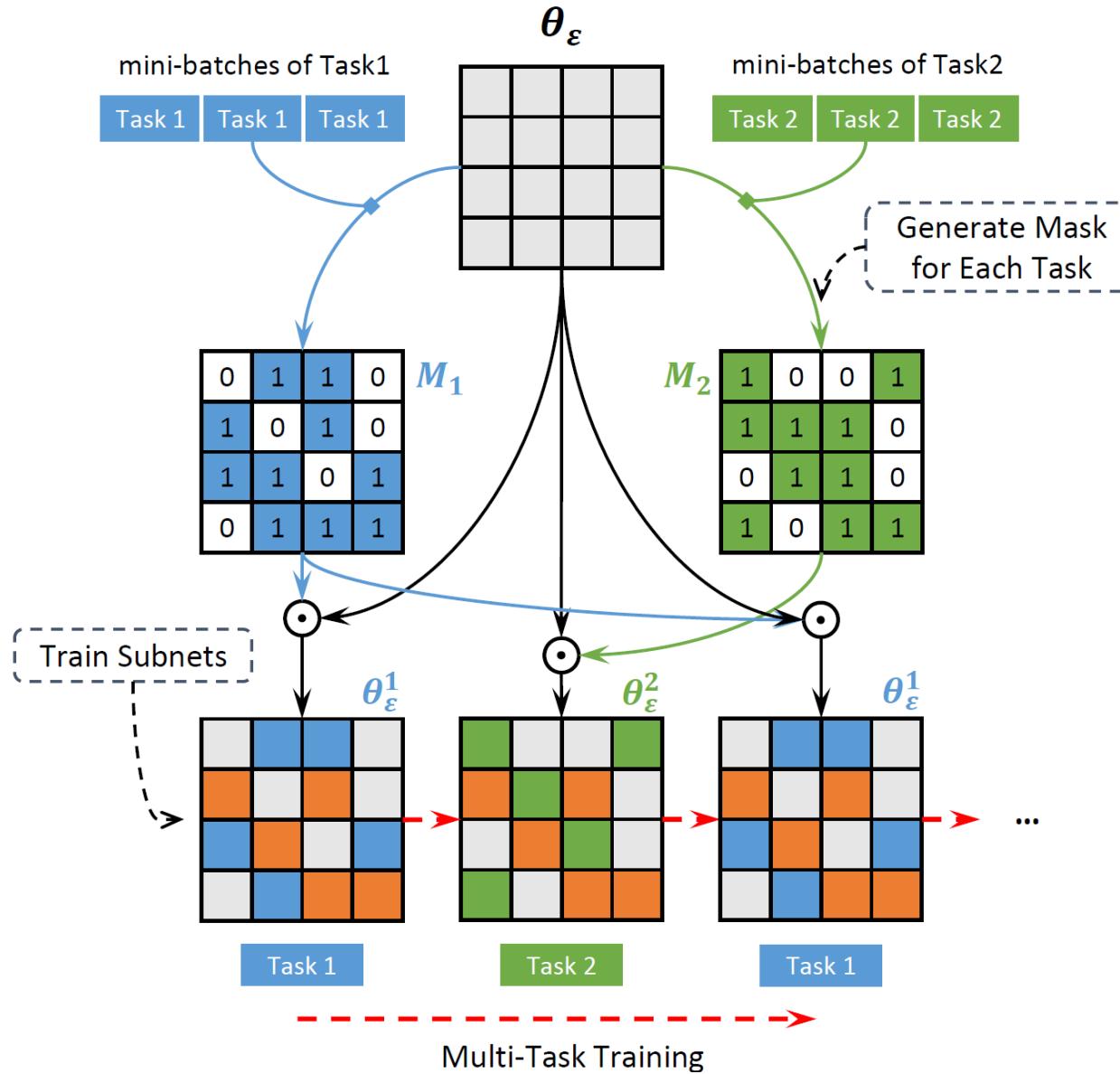
Approach: Learning Sparse Sharing Architectures

Experiments

Analysis and Discussions

Conclusion

# Overview of Our Approach





# Generating Subnets for Each Task

## ➤ Iterative Magnitude Pruning (IMP)

proposed in ([Frankle and Carbin 2019](#)) (ICLR'2019 best paper)

1. Randomly initialize a neural network  $f(x; \theta_0)$  (where  $\theta_0 \sim \mathcal{D}_\theta$ ).
2. Train the network for  $j$  iterations, arriving at parameters  $\theta_j$ .
3. Prune  $p\%$  of the parameters in  $\theta_j$ , creating a mask  $m$ .
4. Reset the remaining parameters to their values in  $\theta_0$ , creating the winning ticket  $f(x; m \odot \theta_0)$ .

# Generating Subnets for Each Task



## ➤ Iterative Magnitude Pruning (IMP)

---

### Algorithm 1 Sparse Sharing Architecture Learning

---

**Require:** Base Network  $\mathcal{E}$ ; Pruning rate  $\alpha$ ; Minimal sparsity  $S$ ; Datasets for  $T$  tasks  $\mathcal{D}_1, \dots, \mathcal{D}_T$ , where  $\mathcal{D}_t = \{x_n^t, y_n^t\}_{n=1}^{N_t}$ .

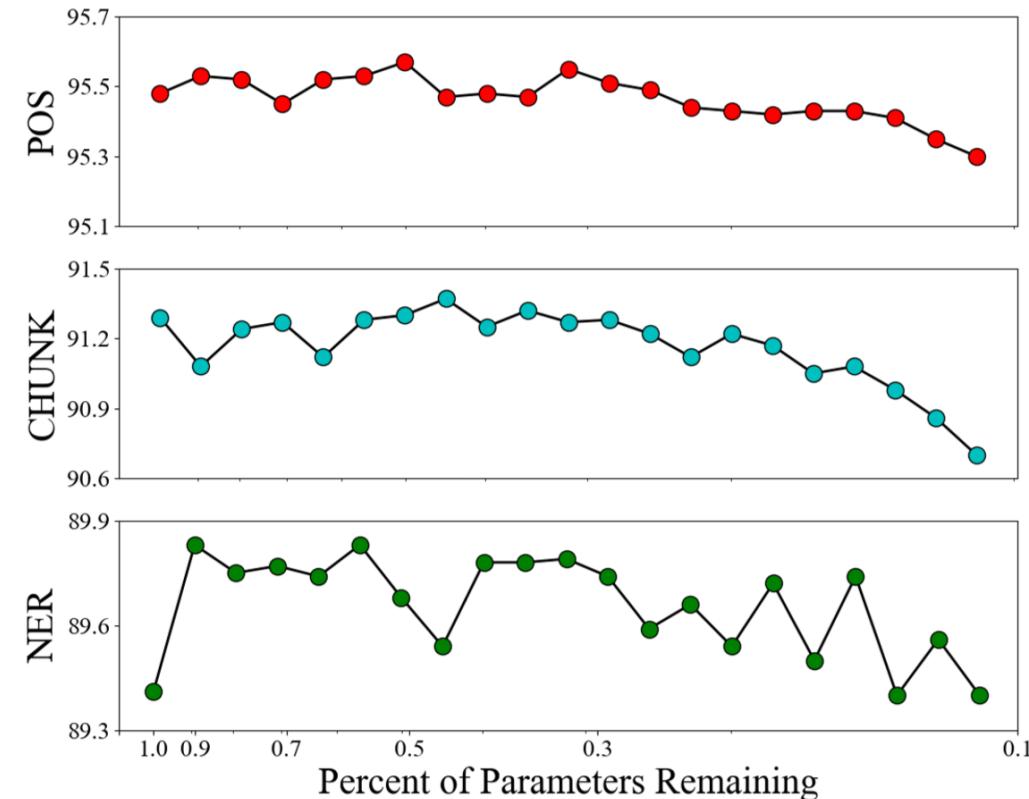
- 1: Randomly initialize  $\theta_{\mathcal{E}}$  to  $\theta_{\mathcal{E}}^{(0)}$ .
  - 2: **for**  $t = 1 \dots T$  **do**
  - 3:   Initialize mask  $M_t^z = \mathbf{1}^{|\theta_{\mathcal{E}}|}$ , where  $z = 1$ .
  - 4:   Train  $\mathcal{E}(x; M_t^z \odot \theta_{\mathcal{E}})$  for  $k$  steps with data sampled from  $\mathcal{D}_t$ , producing network  $\mathcal{E}(x; M_t^z \odot \theta_{\mathcal{E}}^{(k)})$ . Let  $z \leftarrow z + 1$ .
  - 5:   Prune  $\alpha$  percent of the remaining parameters with the lowest magnitudes from  $\theta_{\mathcal{E}}^{(k)}$ . That is, let  $M_t^z[j] = 0$  if  $\theta_{\mathcal{E}}^{(k)}[j]$  is pruned.
  - 6:   If  $\frac{\|M_t^z\|_0}{|\theta_{\mathcal{E}}|} \leq S$ , the masks for task  $t$  are  $\{M_t^i\}_{i=1}^z$ .
  - 7:   Otherwise, reset  $\theta_{\mathcal{E}}$  to  $\theta_{\mathcal{E}}^{(0)}$  and repeat steps 4-6 iteratively to learn more sparse subnetwork.
  - 8: **end for**
  - 9: **return**  $\{M_1^i\}_{i=1}^z, \{M_2^i\}_{i=1}^z, \dots, \{M_T^i\}_{i=1}^z$ .
-

# Select Subnets



- Pick the subnet that performs best on the dev set.
- If there are multiple best-performing subnets, take the subnet with the lowest sparsity.

POS	CHUNK	NER
50.12%	44.67%	56.23%



# Training Subnets in Parallel

1. Select the next task  $t$ .
  - Proportional sampling ([Sanh, Wolf, and Ruder 2019](#))
2. Select a random mini-batch for task  $t$ .
3. Feed this batch of data into the subnetwork corresponding to task  $t$ , i.e.  $\mathcal{E}(x; M_t \odot \theta_{\mathcal{E}})$ .
4. Update the subnetwork parameters for this task by taking a gradient step with respect to this mini-batch.
5. Go to 1.

$$\mathcal{L}(\theta) = \sum_{t=1}^T \lambda_t \sum_{n=1}^{N_t} \mathcal{L}_t(\hat{y}_n^t, y_n^t)$$

# Multi-Task Warmup (MTW)



---

**Algorithm 1** Sparse Sharing Architecture Learning

---

**Require:** Base Network  $\mathcal{E}$ ; Pruning rate  $\alpha$ ; Minimal sparsity  $S$ ; Datasets for  $T$  tasks  $\mathcal{D}_1, \dots, \mathcal{D}_T$ , where  $\mathcal{D}_t = \{x_n^t, y_n^t\}_{n=1}^{N_t}$ .

- 1: Randomly initialize  $\theta_{\mathcal{E}}$  to  $\theta_{\mathcal{E}}^{(0)}$ . MTW:  $\theta_{\mathcal{E}}^{(0)} \rightarrow \theta_{\mathcal{E}}^{(w)}$
  - 2: **for**  $t = 1 \dots T$  **do**
  - 3:   Initialize mask  $M_t^z = \mathbf{1}^{|\theta_{\mathcal{E}}|}$ , where  $z = 1$ .
  - 4:   Train  $\mathcal{E}(x; M_t^z \odot \theta_{\mathcal{E}})$  for  $k$  steps with data sampled from  $\mathcal{D}_t$ , producing network  $\mathcal{E}(x; M_t^z \odot \theta_{\mathcal{E}}^{(k)})$ . Let  $z \leftarrow z + 1$ .
  - 5:   Prune  $\alpha$  percent of the remaining parameters with the lowest magnitudes from  $\theta_{\mathcal{E}}^{(k)}$ . That is, let  $M_t^z[j] = 0$  if  $\theta_{\mathcal{E}}^{(k)}[j]$  is pruned.
  - 6:   If  $\frac{\|M_t^z\|_0}{|\theta_{\mathcal{E}}|} \leq S$ , the masks for task  $t$  are  $\{M_t^i\}_{i=1}^z$ .
  - 7:   Otherwise, reset  $\theta_{\mathcal{E}}$  to  $\theta_{\mathcal{E}}^{(w)}$  and repeat steps 4-6 iteratively to learn more sparse subnetwork.
  - 8: **end for**
  - 9: **return**  $\{M_1^i\}_{i=1}^z, \{M_2^i\}_{i=1}^z, \dots, \{M_T^i\}_{i=1}^z$ .
-

# Outline



Multi-Task Learning: Sharing Mechanisms

Sparse Sharing Mechanism

Approach: Learning Sparse Sharing Architectures

Experiments

Analysis and Discussions

Conclusion

# Experiments



➤ Tasks: Part-of-Speech, NER, Chunking

➤ Datasets

Exp1: CoNLL-2003

Exp2: OntoNotes 5.0

Exp3: PTB + CoNLL-2003 + CoNLL-2000

➤ Model Settings

Base model: CNN-BiLSTM ([Ma and Hovy 2016](#))

Multi-Task baselines: hard/soft/hierarchical sharing

# Exp1 & Exp2



Systems	POS		NER		Chunking		# Params
	Test Acc.	Δ	Test F1	Δ	Test F1	Δ	
Exp1: CoNLL-2003							
Single task	95.09	-	89.36	-	89.92	-	1602k
Single task (subnet)	95.11	+0.02	89.39	+0.03	89.96	+0.04	811k
Hard sharing	95.34	+0.25	88.68	-0.68	90.92	+1.00	534k
Soft sharing	95.16	+0.07	89.35	-0.01	90.71	+0.79	1596k
Hierarchical sharing	95.09	+0.00	89.30	-0.06	90.89	+0.97	1497k
<b>Sparse sharing (ours)</b>	<b>95.56</b>	<b>+0.47</b>	<b>90.35</b>	<b>+0.99</b>	<b>91.55</b>	<b>+1.63</b>	<b>396k</b>
Exp2: OntoNotes 5.0							
Single task	97.40	-	82.72	-	95.21	-	4491k
Single task (subnet)	97.42	+0.02	82.94	+0.22	95.28	+0.07	1459k
Hard sharing	97.46	+0.06	82.95	+0.23	95.52	+0.31	1497k
Soft sharing	97.34	-0.06	81.93	-0.79	95.29	+0.08	4485k
Hierarchical sharing	97.22	-0.18	82.81	+0.09	95.53	+0.32	1497k
<b>Sparse sharing (ours)</b>	<b>97.54</b>	<b>+0.14</b>	<b>83.42</b>	<b>+0.70</b>	<b>95.56</b>	<b>+0.35</b>	<b>662k</b>

# Exp1 & Exp2



Systems	POS		NER		Chunking		# Params
	Test Acc.	△	Test F1	△	Test F1	△	
Exp1: CoNLL-2003							
Single task	95.09	-	89.36	-	89.92	-	1602k
Single task (subnet)	95.11	+0.02	89.39	+0.03	89.96	+0.04	811k
Hard sharing	95.34	+0.25	88.68	-0.68	90.92	+1.00	534k
Soft sharing	95.16	+0.07	89.35	-0.01	90.71	+0.79	1596k
Hierarchical sharing	95.09	+0.00	89.30	-0.06	90.89	+0.97	1497k
<b>Sparse sharing (ours)</b>	<b>95.56</b>	+0.47	<b>90.35</b>	+0.99	<b>91.55</b>	+1.63	<b>396k</b>
Exp2: OntoNotes 5.0							
Single task	97.40	-	82.72	-	95.21	-	4491k
Single task (subnet)	97.42	+0.02	82.94	+0.22	95.28	+0.07	1459k
Hard sharing	97.46	+0.06	82.95	+0.23	95.52	+0.31	1497k
Soft sharing	97.34	-0.06	81.93	-0.79	95.29	+0.08	4485k
Hierarchical sharing	97.22	-0.18	82.81	+0.09	95.53	+0.32	1497k
<b>Sparse sharing (ours)</b>	<b>97.54</b>	+0.14	<b>83.42</b>	+0.70	<b>95.56</b>	+0.35	<b>662k</b>

- Compared with results reported in previous work
- Equipped with CRF

Systems	POS	NER	Chunk.
<i>Single Task Models:</i>			
(Collobert et al. 2011)	97.29	89.59	<b>94.32</b>
(Huang, Xu, and Yu 2015)	97.25	89.91	93.67
(Chen et al. 2018)	<b>97.30</b>	<b>90.08</b>	93.71
<i>Multi-Task Models:</i>			
Hard sharing†	97.23	90.38	94.32
Meta-MTL-LSTM†	97.45	90.72	95.11
<b>Sparse sharing (ours)</b>	<b>97.57</b>	<b>90.87</b>	<b>95.26</b>

# Outline



Multi-Task Learning: Sharing Mechanisms

Sparse Sharing Mechanism

Approach: Learning Sparse Sharing Architectures

Experiments

Analysis and Discussions

Conclusion

# About Negative Transfer



In the future studies, there are several issues to be addressed. Firstly, outlier tasks, which are unrelated to other tasks, are well known to hamper the performance of all the tasks when learning them jointly. There are some methods to alleviate negative effects outlier tasks bring. However, there lacks principled ways and theoretical analyses to study the resulting negative effects. In order to make MTL safe to be used by human, this is an important issue and needs more studies.

# About Negative Transfer



- Construct a loosely related multi-task setting
  - Real: Named Entity Recognition (NER)
  - Synthetic: Position Prediction (PP)

	NER	$\Delta$	PP	$\Delta$
Single task	71.05	-	99.21	-
Hard sharing	61.62	-9.43	99.50	+0.29
Sparse sharing	71.46	+0.41	99.45	+0.24

# About Task Relatedness



- Define mask overlap ratio (OR) as:

$$\text{OR}(M_1, M_2, \dots, M_T) = \frac{\|\cap_{t=1}^T M_t\|_0}{\|\cup_{t=1}^T M_t\|_0}$$

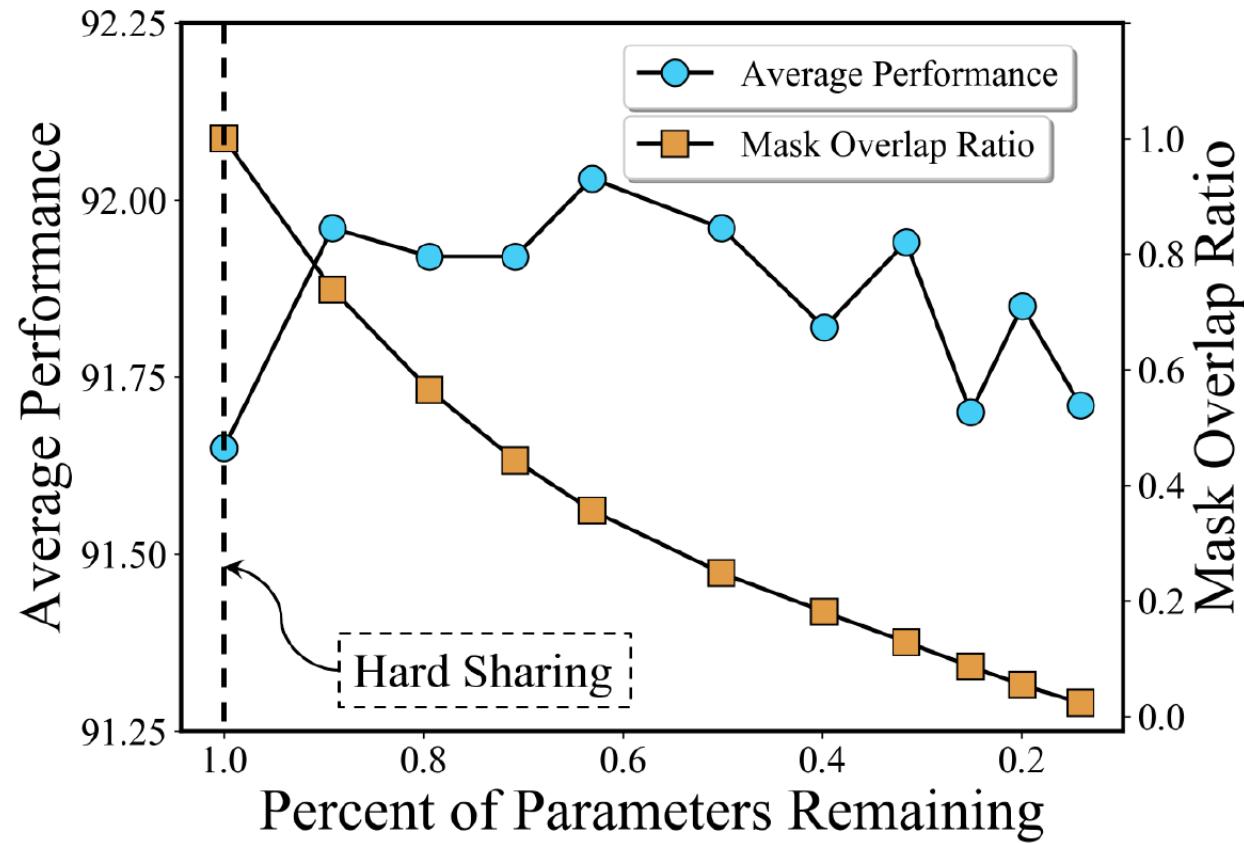
Task Pairs	Mask OR	$\Delta(S^2 - HS)$
POS & NER	0.18	0.4
NER & Chunking	0.20	0.34
POS & Chunking	0.50	0.05

Table 7: Mask Overlap Ratio (OR) and the improvement for sparse sharing ( $S^2$ ) compared to hard sharing ( $HS$ ) of tasks on CoNLL-2003. The improvement is calculated using the average performance on the test set.

# About Sparsity



- Combinations of subnets with different sparsity



# About Multi-Task Warmup: An ablation study



- Stabilize subnets ([Frankle et al. 2019](#))

	POS	NER	Chunking
CoNLL-2003			
Sparse sharing	95.56	90.35	91.55
– MTW	95.36	89.62	91.04
OntoNotes 5.0			
Sparse sharing	97.54	83.42	95.56
– MTW	97.53	81.15	95.48

# Outline



Multi-Task Learning: Sharing Mechanisms

Sparse Sharing Mechanism

Approach: Learning Sparse Sharing Architectures

Experiments

Analysis and Discussions

Conclusion

# Conclusion



- Does sparse sharing architecture meets the requirements?
  1. It is compatible with a wide range of tasks, regardless of whether the tasks are related or not.
  2. It does not depend on manually designing the sharing structure based on characteristic of tasks.
  3. It is parameter efficient.
- It seems **YES!**



復旦大學



Thanks !

Q & A

[txsun19@fudan.edu.cn](mailto:txsun19@fudan.edu.cn)

# Sequence Labeling Tasks



## ➤ POS, NER and Chunking

Words	Results	of	South	Korean
POS	NNS	IN	JJ	JJ
NER	O	O	B-MISC	I-MISC
Chunk.	B-NP	B-PP	B-NP	I-NP



# CNN-BiLSTM: A popular architecture in sequence labeling tasks

