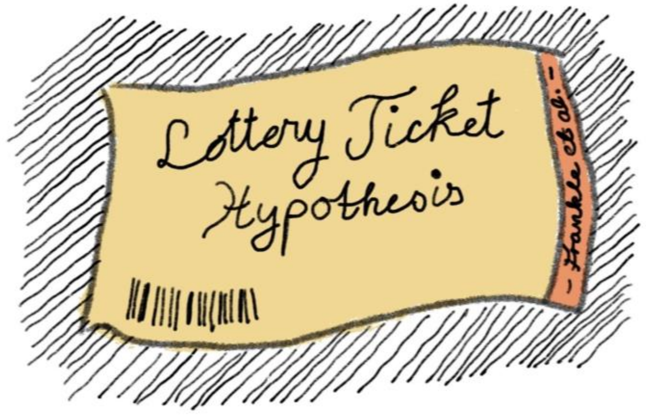
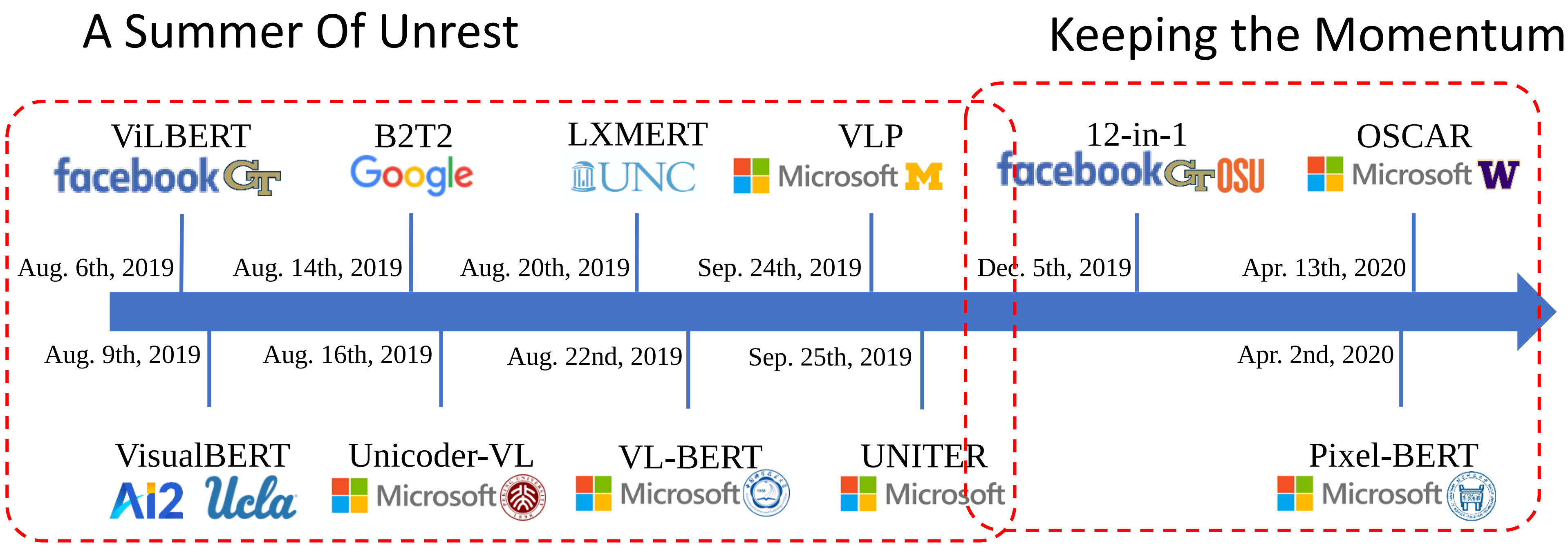
Playing Lottery Tickets with Vision and Language



Zhe Gan

4/9/2021

Great success of VLP models



Downstream Tasks

- VQA
- VCR
- NLVR2
- Visual Entailment
- Referring Expressions
- Image-Text Retrieval
- Image Captioning

Many more models have come out since then: VILLA, VIVO, VinVL etc.

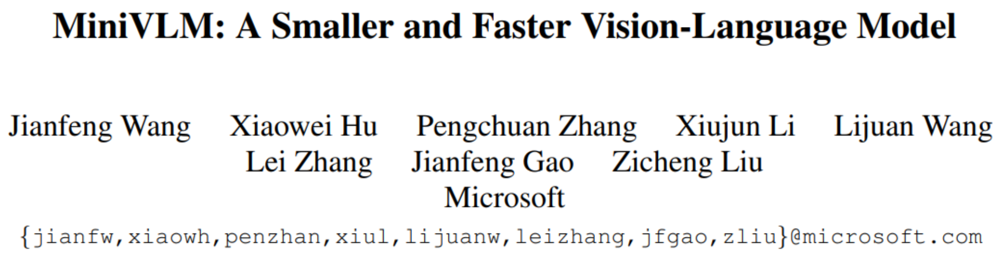
How about efficiency?

VLM Compression

- This topic has not been well investigated yet:
  - Pruning: study the redundancy/over-parameterization of pre-trained weights
  - Quantization: quantifying the continuous weights into discrete values
  - Knowledge Distillation: transferring knowledge from a large teacher to a smaller student
  - Parameter Sharing: such as ALBERT
- Efficient Training:
  - Efficient Transformer: Reformer, Performer, Longformer, BigBird
  - ConvBERT: Convolution + Self-Attention

Can we prune a large VLP model while preserving its performance and transferability?

Existing/Co-current Work



MiniVLM: A Smaller and Faster Vision-Language Model

Jianfeng Wang Xiaowei Hu Pengchuan Zhang Xiujun Li Lijuan Wang
Lei Zhang Jianfeng Gao Zicheng Liu
Microsoft
{jianfw,xiaowh,penzhan,xiul,lijuanw,leizhang,jfgao,zliu}@microsoft.com

- MiniVLM: A Smaller and Faster Vision-Language Model

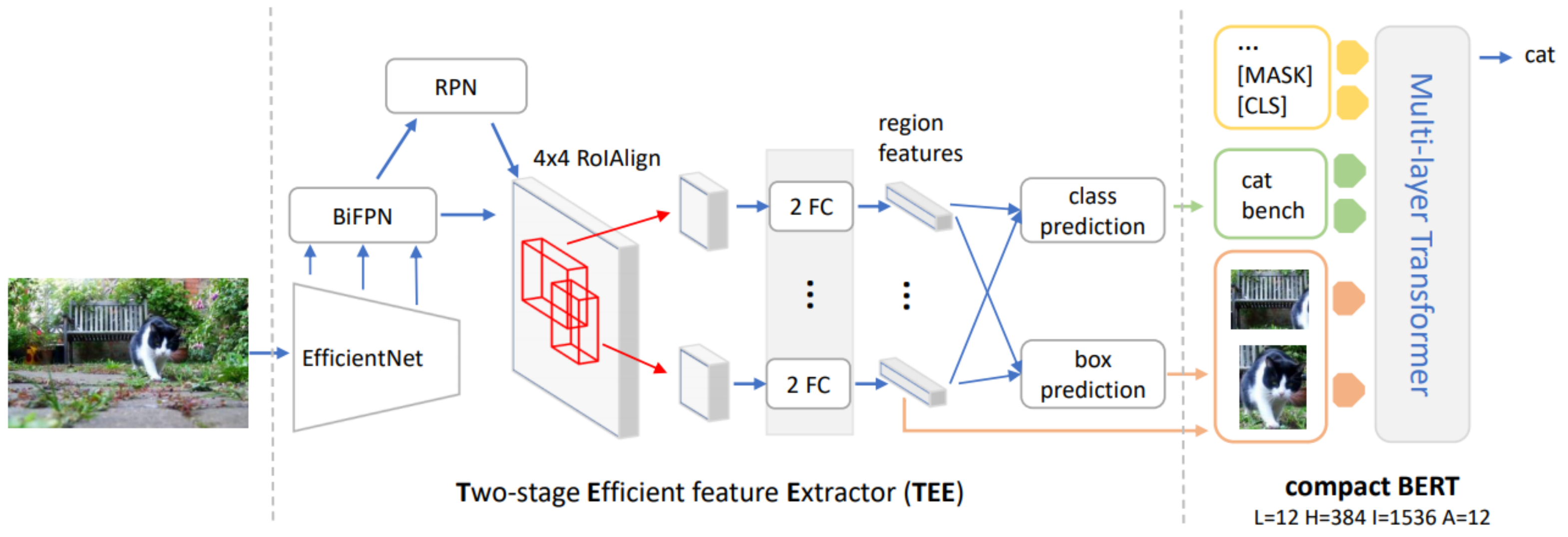
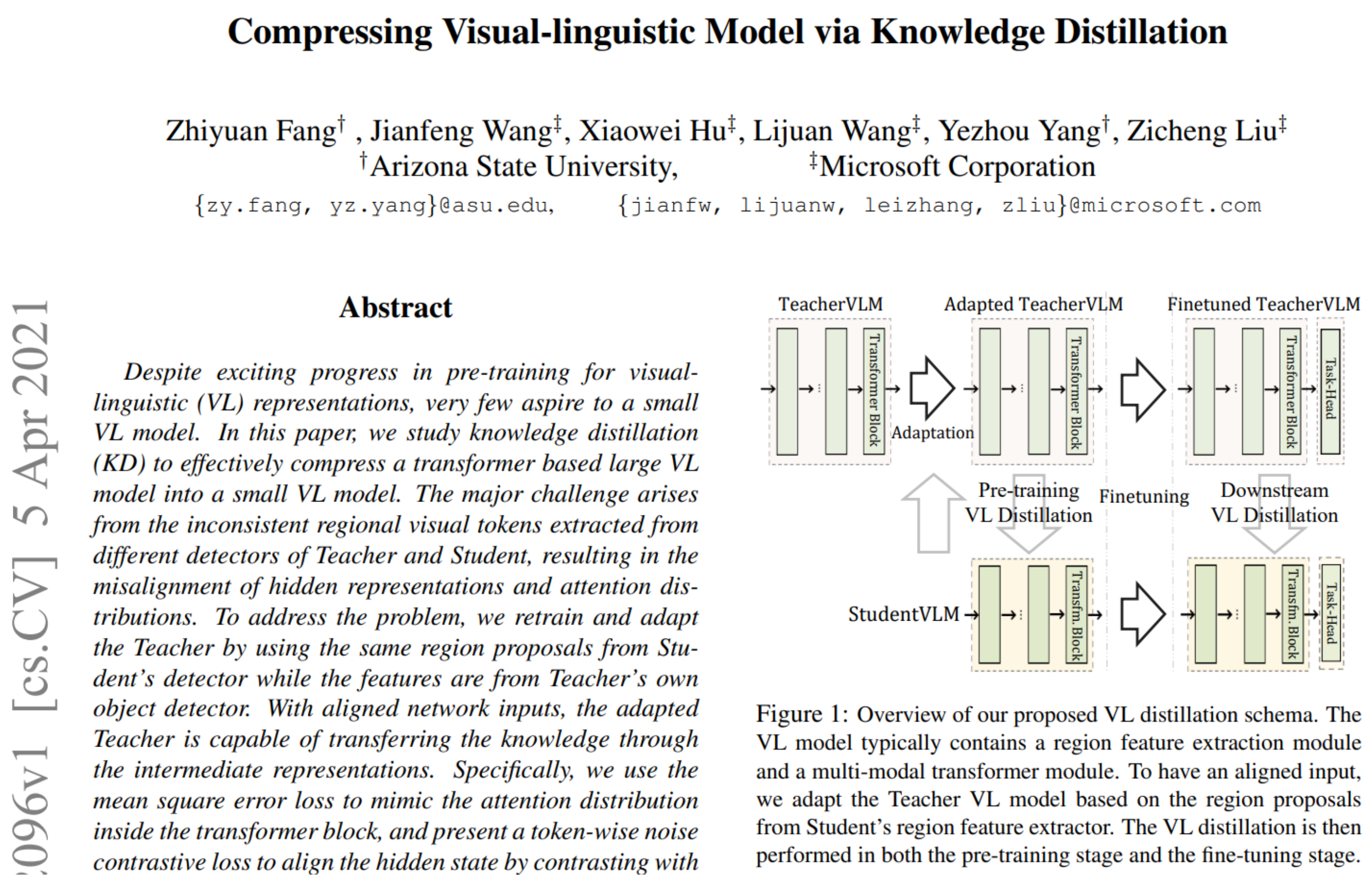


Figure 2: The proposed MiniVLM architecture, consisting of the Two-stage Efficient feature Extractor (TEE) and compact BERT feature fusion module. During inference, the vision module detects objects and extracts region features, which are fed to the transformer model. The text input of the transformer also depends on the downstream task. For image captioning, [CLS] is given as the first token, then the model predicts the next token in an auto-regressive manner to generate a caption sentence.

Existing/Co-current Work

- DistilVLM: Compressing Visual-linguistic Model via Knowledge Distillation



Compressing Visual-linguistic Model via Knowledge Distillation

Zhiyuan Fang[†] , Jianfeng Wang[‡], Xiaowei Hu[‡], Lijuan Wang[‡], Yezhou Yang[†], Zicheng Liu[‡]
[†]Arizona State University, [‡]Microsoft Corporation
{zy.fang, yz.yang}@asu.edu, {jianfw, lijuanw, leizhang, zliu}@microsoft.com

Abstract

Despite exciting progress in pre-training for visual-linguistic (VL) representations, very few aspire to a small VL model. In this paper, we study knowledge distillation (KD) to effectively compress a transformer based large VL model into a small VL model. The major challenge arises from the inconsistent regional visual tokens extracted from different detectors of Teacher and Student, resulting in the misalignment of hidden representations and attention distributions. To address the problem, we retrain and adapt the Teacher by using the same region proposals from Student's detector while the features are from Teacher's own object detector. With aligned network inputs, the adapted Teacher is capable of transferring the knowledge through the intermediate representations. Specifically, we use the mean square error loss to mimic the attention distribution inside the transformer block, and present a token-wise noise contrastive loss to align the hidden state by contrasting with

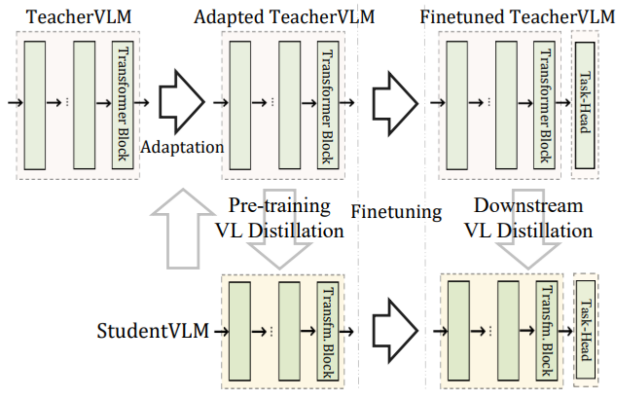


Figure 1: Overview of our proposed VL distillation schema. The VL model typically contains a region feature extraction module and a multi-modal transformer module. To have an aligned input, we adapt the Teacher VL model based on the region proposals from Student's region feature extractor. The VL distillation is then performed in both the pre-training stage and the fine-tuning stage.

Pruning in Deep Learning

- Motivation:
  - It supports generalization by regularizing overparametrized functions.
  - It reduces the memory constraints during inference time by identifying well-performing smaller networks which can fit in memory.
  - It reduces energy costs, computations, storage and latency which can all support deployment on mobile devices.

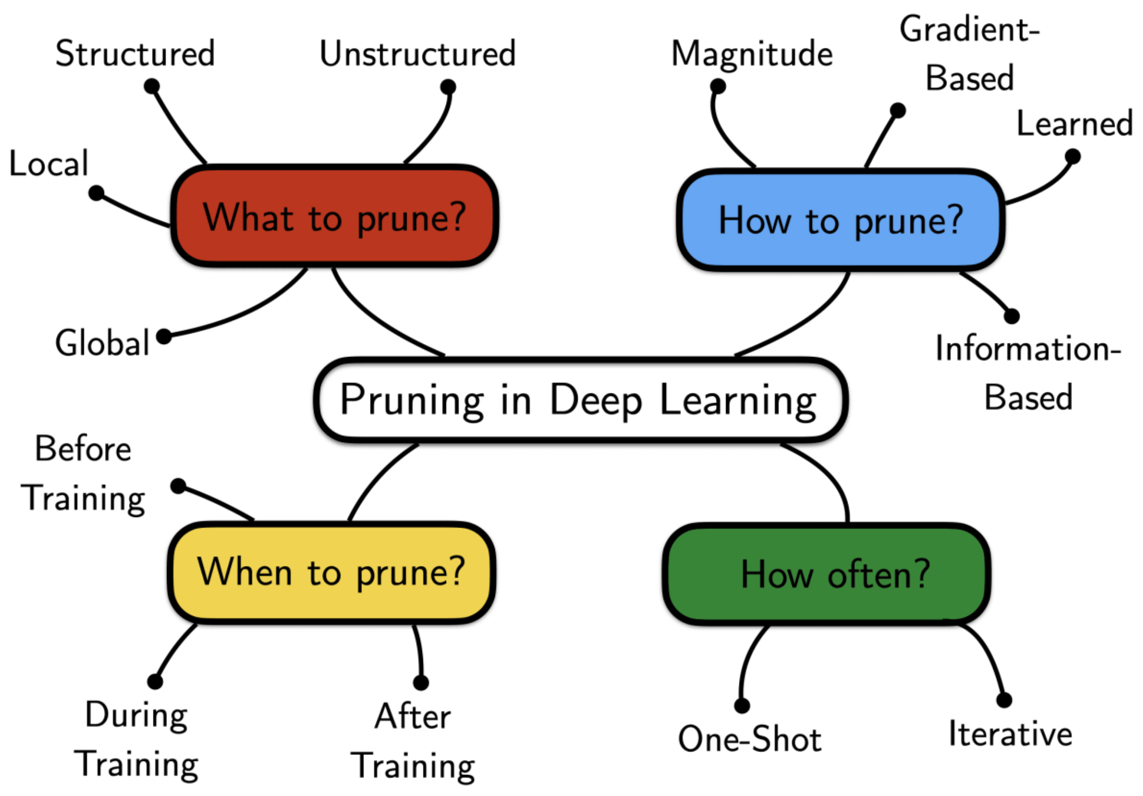


Image source: https://roberttlange.github.io/posts/2020/06/lottery-ticket-hypothesis/

Pruning in Deep Learning

- In this work, we focus on using the *lottery ticket hypothesis* to study the over-parameterization of VLP models

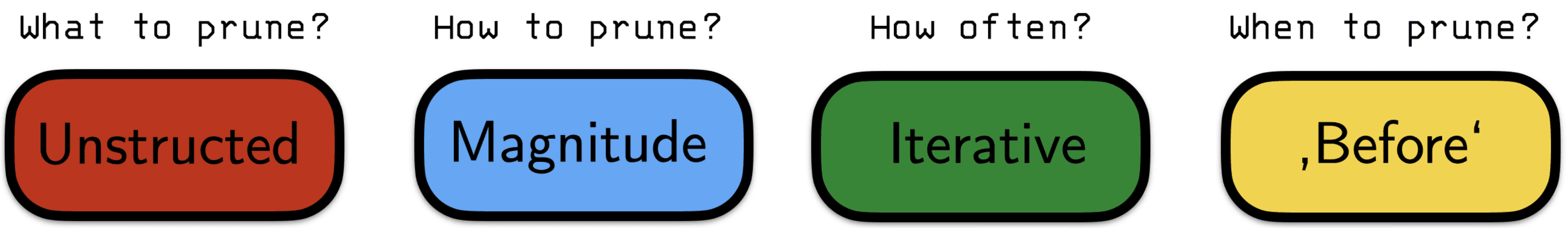
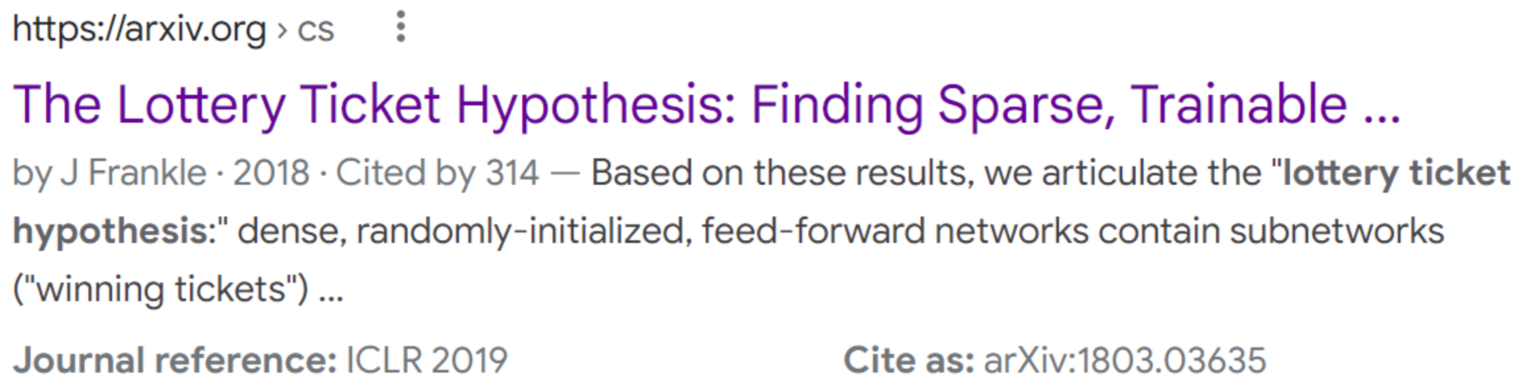


Image source: https://roberttlange.github.io/posts/2020/06/lottery-ticket-hypothesis/

Lottery Ticket Hypothesis

- ICLR 2019 Best Paper by MIT: The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks
- LTH: A randomly initialized, dense neural network contains a subnetwork that is initialized such that — when trained in isolation — it can match the test accuracy of the original network after training for at most the same number of iterations. - Frankle & Carbin (2019, p.2)
- An emerging sub-field in deep learning regarding sparse neural networks


https://arxiv.org › cs

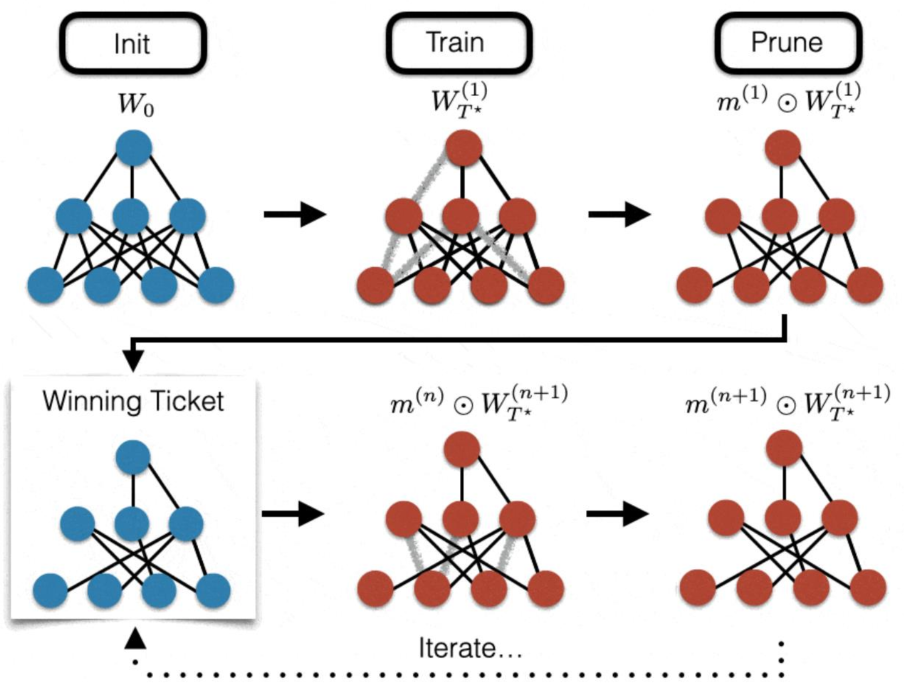
The Lottery Ticket Hypothesis: Finding Sparse, Trainable ...

by J Frankle · 2018 · Cited by 314 — Based on these results, we articulate the "**lottery ticket hypothesis:**" dense, randomly-initialized, feed-forward networks contain subnetworks ("winning tickets") ...

Journal reference: ICLR 2019 **Cite as:** arXiv:1803.03635

Lottery Ticket Hypothesis

- Find the lottery ticket via unstructured IMP (Iterative Magnitude Pruning)



Init
W_0
Train
$W_{T^\star}^{(1)}$
Prune
$m^{(1)} \odot W_{T^\star}^{(1)}$
Winning Ticket
$m^{(n)} \odot W_{T^\star}^{(n+1)}$
$m^{(n+1)} \odot W_{T^\star}^{(n+1)}$
Iterate…

- Stabilizing the Lottery Ticket Hypothesis via *Rewinding*: making LTH work for large CNN backbones
- Many other papers, to name a few:
  - Comparing Rewinding and Fine-tuning in Neural Network Pruning, ICLR 2020
  - Deconstructing lottery tickets: Zeros, signs, and the supermask, NeurIPS 2019
  - Linear Mode Connectivity & the Lottery Ticket Hypothesis, ICML 2020
  - The Early Phase of Neural Network Training, ICLR 2020
  - Drawing Early-Bird Tickets: Towards more efficient training of deep networks, ICLR 2020
  - Pruning neural networks without any data by iteratively conserving synaptic flow, NeurIPS 2020

How Generalizable the LTH is?

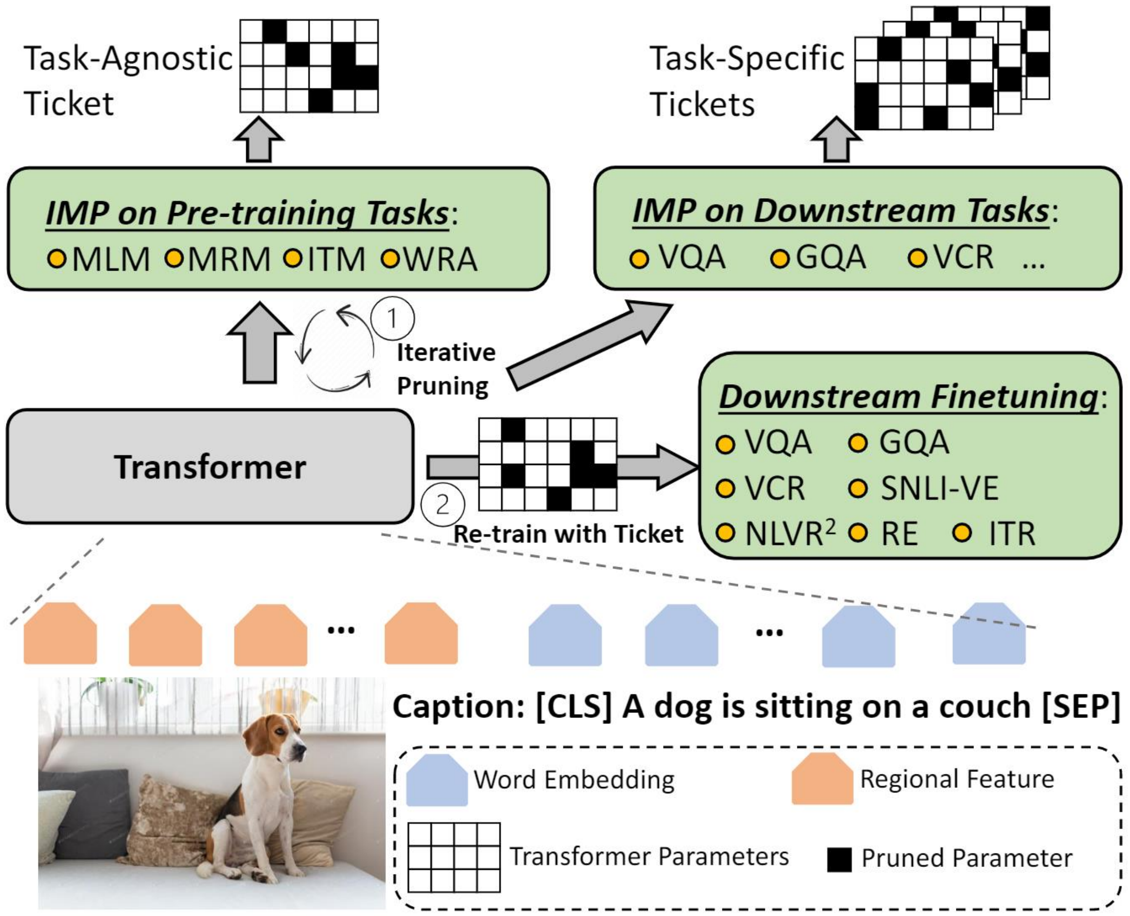
- One ticket to win them all: Generalizing lottery ticket initializations across datasets and optimizers, NeurIPS 2019
- Playing the lottery with rewards and multiple languages: lottery tickets in RL and NLP, ICLR 2020
- When BERT Plays the Lottery, All Tickets Are Winning, EMNLP 2020
- The Lottery Ticket Hypothesis for Pre-trained BERT Networks, NeurIPS 2020
- EarlyBERT: Efficient BERT Training via Early-bird Lottery Tickets, under submission
- GANs Can Play Lottery Tickets Too, ICLR 2021
- Long Live the Lottery: The Existence of Winning Tickets in Lifelong Learning, ICLR 2021

There is no work on VLP lottery tickets yet!

Questions We Aim to Answer

- *Existence*: Can we draw VLP winning tickets successfully for VL downstream tasks?
- *Transferability*: Can we find tickets that transfer universally to all downstream VL tasks?
- *Compatibility*: Can we find tickets compatible with adversarial training to enhance the performance?

Playing Lottery Tickets with Vision and Language



Algorithm 1 Iterative Magnitude Pruning for V+L Tickets.

```
Input  Initial mask $\boldsymbol{m} = 1^{d_1}$; Pre-trained parameters $\boldsymbol{\theta}_0$ and task-specific parameters $\boldsymbol{\phi}_0$; rewinding step $i$ (could be 0), sparsity level $s$, total training step $t$.
Train the pre-trained V+L model $f(\boldsymbol{x}; \boldsymbol{m} \odot \boldsymbol{\theta}_0, \boldsymbol{\phi}_0)$ to step $i$: $f(\boldsymbol{x}; \boldsymbol{m} \odot \boldsymbol{\theta}_i, \boldsymbol{\phi}_i)$.
repeat
    Train $f(\boldsymbol{x}; \boldsymbol{m} \odot \boldsymbol{\theta}_i, \boldsymbol{\phi}_i)$ to step $t$: $f(\boldsymbol{x}; \boldsymbol{m} \odot \boldsymbol{\theta}_t, \boldsymbol{\phi}_t)$.
    Prune $10\%$ of non-zero weights of $\boldsymbol{m} \odot \boldsymbol{\theta}_t$ based on the magnitudes and update $\boldsymbol{m}$ accordingly.
until the sparsity of $\boldsymbol{m}$ reaches $s$
Return  $f(\boldsymbol{x}; \boldsymbol{m} \odot \boldsymbol{\theta}_i, \cdot)$ =0
```

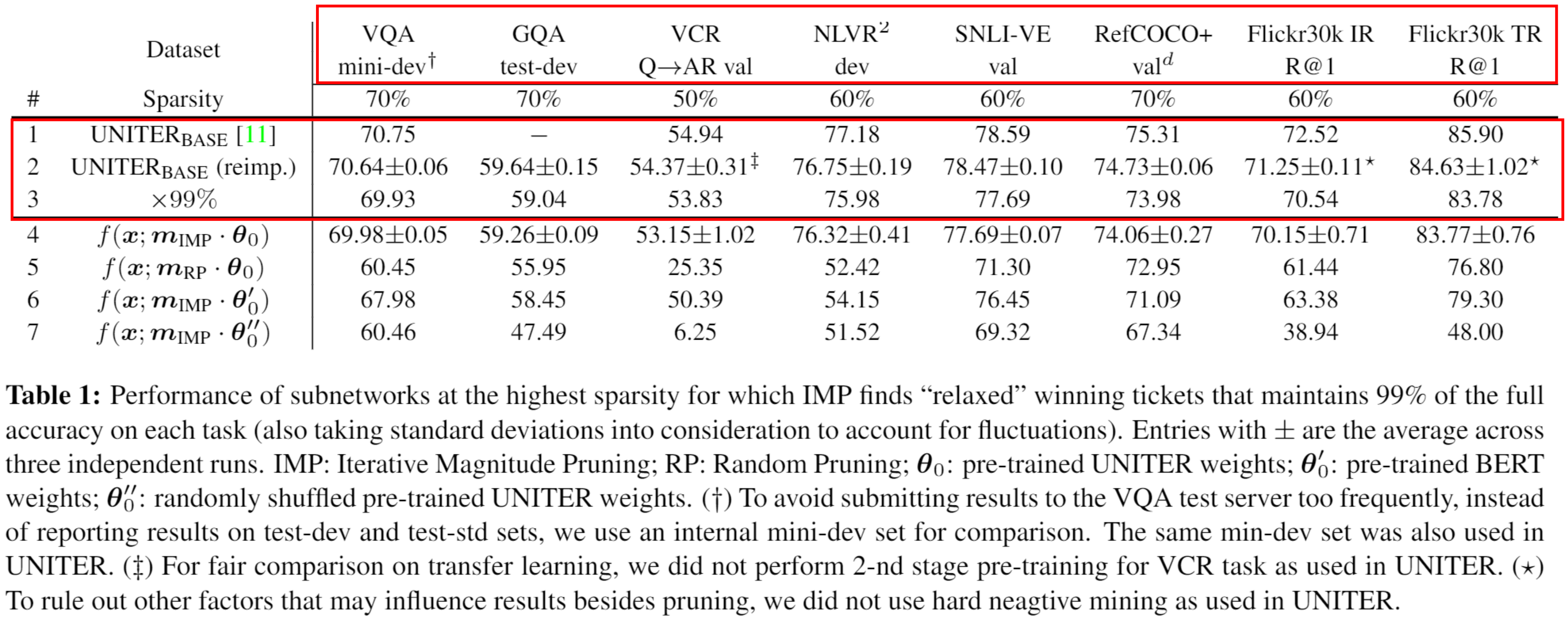
A *winning ticket* is a sub-network that matches the performance of the original full dense network

Our Findings

- ~~*Existence*: Can we draw VLP winning tickets successfully for VL downstream tasks?~~
- *VLM can play lottery tickets too*: We confirm that "relaxed" winning tickets that match 99% of the full accuracy can be found at 50%-70% sparsity across all the tasks.
- ~~*Transferability*: Can we find tickets that transfer universally to all downstream VL tasks?~~
- *One ticket to win them all*: Matching subnetworks found via IMP on pre-training tasks transfer universally. Unexpectedly, matching subnetworks found via IMP on each downstream task also transfer to other tasks reasonably well.
- ~~*Compatibility*: Can we find tickets compatible with adversarial training to enhance the performance?~~
- *Enhancing tickets with adversarial training*: Though the found winning tickets are sparse neural networks, adversarial training can be still helpful to enhance the performance across all the tasks considered.

UNITER Can Play Lottery Tickets Too

- *Q1: Are there winning tickets in UNITER?*

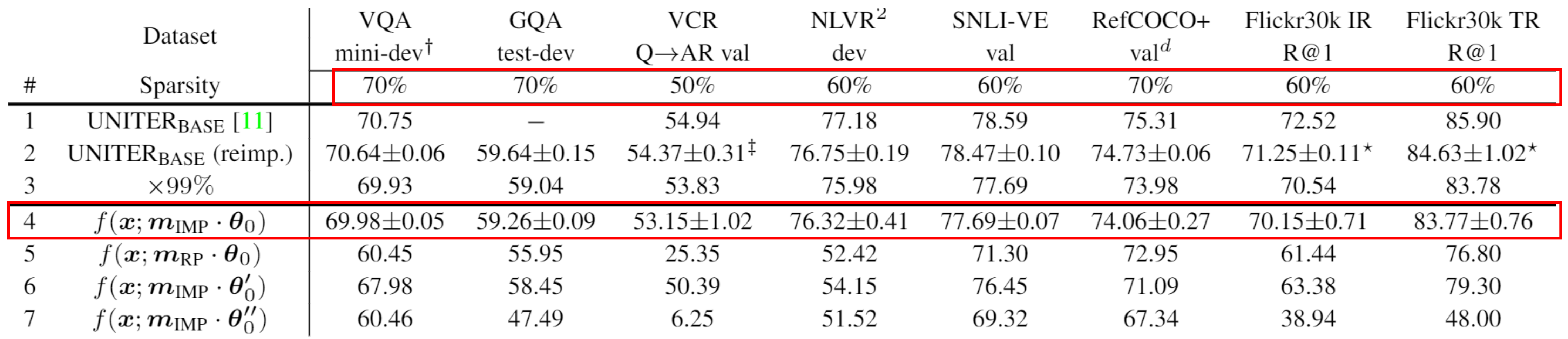


#	Dataset	VQA mini-dev†	GQA test-dev	VCR Q→AR val	NLVR^2 dev	SNLI-VE val	RefCOCO+ val^d	Flickr30k IR R@1	Flickr30k TR R@1
	Sparsity	70%	70%	50%	60%	60%	70%	60%	60%
1	UNITER$_{\text{BASE}}$ [11]	70.75	–	54.94	77.18	78.59	75.31	72.52	85.90
2	UNITER$_{\text{BASE}}$ (reimp.)	70.64±0.06	59.64±0.15	54.37±0.31‡	76.75±0.19	78.47±0.10	74.73±0.06	71.25±0.11⋆	84.63±1.02⋆
3	×99%	69.93	59.04	53.83	75.98	77.69	73.98	70.54	83.78
4	$f(\boldsymbol{x}; \boldsymbol{m}_{\text{IMP}} \cdot \boldsymbol{\theta}_0)$	69.98±0.05	59.26±0.09	53.15±1.02	76.32±0.41	77.69±0.07	74.06±0.27	70.15±0.71	83.77±0.76
5	$f(\boldsymbol{x}; \boldsymbol{m}_{\text{RP}} \cdot \boldsymbol{\theta}_0)$	60.45	55.95	25.35	52.42	71.30	72.95	61.44	76.80
6	$f(\boldsymbol{x}; \boldsymbol{m}_{\text{IMP}} \cdot \boldsymbol{\theta}'_0)$	67.98	58.45	50.39	54.15	76.45	71.09	63.38	79.30
7	$f(\boldsymbol{x}; \boldsymbol{m}_{\text{IMP}} \cdot \boldsymbol{\theta}''_0)$	60.46	47.49	6.25	51.52	69.32	67.34	38.94	48.00

Table 1: Performance of subnetworks at the highest sparsity for which IMP finds "relaxed" winning tickets that maintains 99% of the full accuracy on each task (also taking standard deviations into consideration to account for fluctuations). Entries with ± are the average across three independent runs. IMP: Iterative Magnitude Pruning; RP: Random Pruning; $\boldsymbol{\theta}_0$: pre-trained UNITER weights; $\boldsymbol{\theta}'_0$: pre-trained BERT weights; $\boldsymbol{\theta}''_0$: randomly shuffled pre-trained UNITER weights. (†) To avoid submitting results to the VQA test server too frequently, instead of reporting results on test-dev and test-std sets, we use an internal mini-dev set for comparison. The same min-dev set was also used in UNITER. (‡) For fair comparison on transfer learning, we did not perform 2-nd stage pre-training for VCR task as used in UNITER. (⋆) To rule out other factors that may influence results besides pruning, we did not use hard neagtive mining as used in UNITER.

UNITER Can Play Lottery Tickets Too

- *Q1: Are there winning tickets in UNITER?*

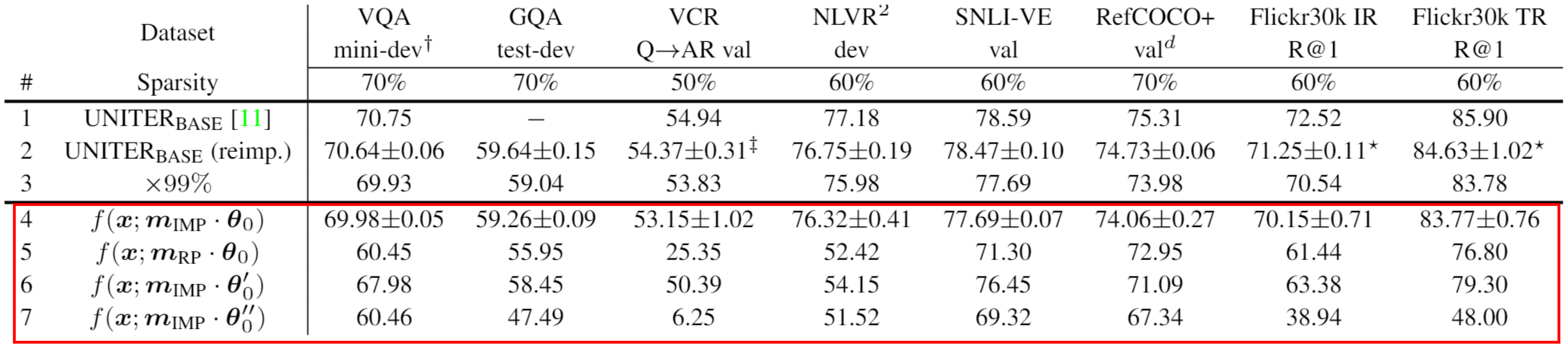


	Dataset	VQA mini-dev[†]	GQA test-dev	VCR Q→AR val	NLVR^2 dev	SNLI-VE val	RefCOCO+ val^d	Flickr30k IR R@1	Flickr30k TR R@1
#	Sparsity	70%	70%	50%	60%	60%	70%	60%	60%
1	UNITER$_{\text{BASE}}$ [11]	70.75	–	54.94	77.18	78.59	75.31	72.52	85.90
2	UNITER$_{\text{BASE}}$ (reimp.)	70.64±0.06	59.64±0.15	54.37±0.31[‡]	76.75±0.19	78.47±0.10	74.73±0.06	71.25±0.11[⋆]	84.63±1.02[⋆]
3	×99%	69.93	59.04	53.83	75.98	77.69	73.98	70.54	83.78
4	$f(\boldsymbol{x}; \boldsymbol{m}_{\text{IMP}} \cdot \boldsymbol{\theta}_0)$	69.98±0.05	59.26±0.09	53.15±1.02	76.32±0.41	77.69±0.07	74.06±0.27	70.15±0.71	83.77±0.76
5	$f(\boldsymbol{x}; \boldsymbol{m}_{\text{RP}} \cdot \boldsymbol{\theta}_0)$	60.45	55.95	25.35	52.42	71.30	72.95	61.44	76.80
6	$f(\boldsymbol{x}; \boldsymbol{m}_{\text{IMP}} \cdot \boldsymbol{\theta}'_0)$	67.98	58.45	50.39	54.15	76.45	71.09	63.38	79.30
7	$f(\boldsymbol{x}; \boldsymbol{m}_{\text{IMP}} \cdot \boldsymbol{\theta}''_0)$	60.46	47.49	6.25	51.52	69.32	67.34	38.94	48.00

Table 1: Performance of subnetworks at the highest sparsity for which IMP finds "relaxed" winning tickets that maintains 99% of the full accuracy on each task (also taking standard deviations into consideration to account for fluctuations). Entries with ± are the average across three independent runs. IMP: Iterative Magnitude Pruning; RP: Random Pruning; $\boldsymbol{\theta}_0$: pre-trained UNITER weights; $\boldsymbol{\theta}'_0$: pre-trained BERT weights; $\boldsymbol{\theta}''_0$: randomly shuffled pre-trained UNITER weights. (†) To avoid submitting results to the VQA test server too frequently, instead of reporting results on test-dev and test-std sets, we use an internal mini-dev set for comparison. The same min-dev set was also used in UNITER. (‡) For fair comparison on transfer learning, we did not perform 2-nd stage pre-training for VCR task as used in UNITER. (⋆) To rule out other factors that may influence results besides pruning, we did not use hard neagtive mining as used in UNITER.

UNITER Can Play Lottery Tickets Too

- *Q2: Are IMP winning tickets sparser than randomly pruned or initialized subnetworks?*

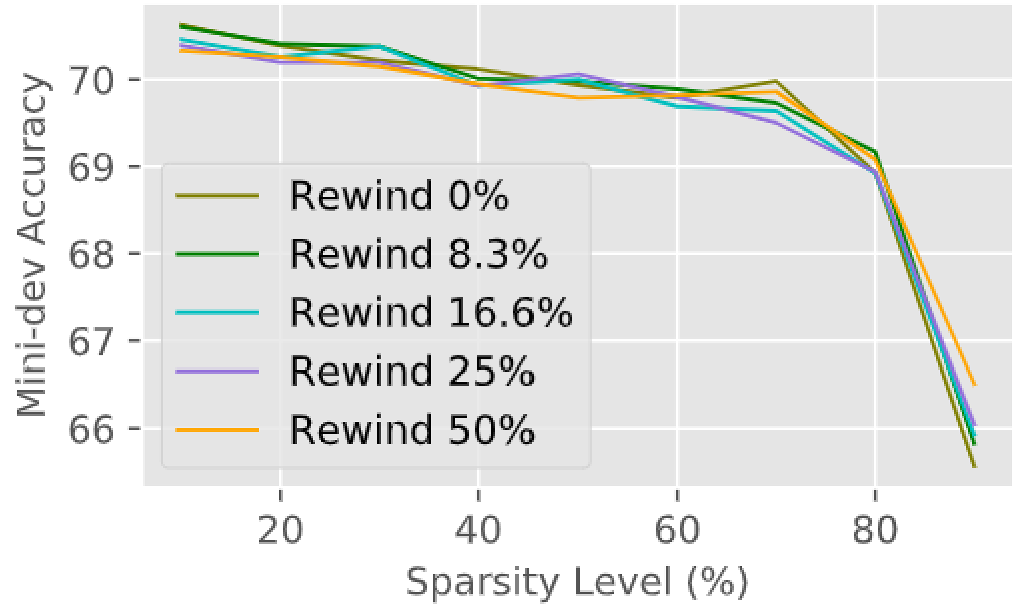


	Dataset	VQA mini-dev[†]	GQA test-dev	VCR Q→AR val	NLVR^2 dev	SNLI-VE val	RefCOCO+ val^d	Flickr30k IR R@1	Flickr30k TR R@1
#	Sparsity	70%	70%	50%	60%	60%	70%	60%	60%
1	UNITER$_{\text{BASE}}$ [11]	70.75	–	54.94	77.18	78.59	75.31	72.52	85.90
2	UNITER$_{\text{BASE}}$ (reimp.)	70.64±0.06	59.64±0.15	54.37±0.31[‡]	76.75±0.19	78.47±0.10	74.73±0.06	71.25±0.11[⋆]	84.63±1.02[⋆]
3	×99%	69.93	59.04	53.83	75.98	77.69	73.98	70.54	83.78
4	$f(\boldsymbol{x}; \boldsymbol{m}_{\text{IMP}} \cdot \boldsymbol{\theta}_0)$	69.98±0.05	59.26±0.09	53.15±1.02	76.32±0.41	77.69±0.07	74.06±0.27	70.15±0.71	83.77±0.76
5	$f(\boldsymbol{x}; \boldsymbol{m}_{\text{RP}} \cdot \boldsymbol{\theta}_0)$	60.45	55.95	25.35	52.42	71.30	72.95	61.44	76.80
6	$f(\boldsymbol{x}; \boldsymbol{m}_{\text{IMP}} \cdot \boldsymbol{\theta}'_0)$	67.98	58.45	50.39	54.15	76.45	71.09	63.38	79.30
7	$f(\boldsymbol{x}; \boldsymbol{m}_{\text{IMP}} \cdot \boldsymbol{\theta}''_0)$	60.46	47.49	6.25	51.52	69.32	67.34	38.94	48.00

Table 1: Performance of subnetworks at the highest sparsity for which IMP finds "relaxed" winning tickets that maintains 99% of the full accuracy on each task (also taking standard deviations into consideration to account for fluctuations). Entries with ± are the average across three independent runs. IMP: Iterative Magnitude Pruning; RP: Random Pruning; $\boldsymbol{\theta}_0$: pre-trained UNITER weights; $\boldsymbol{\theta}'_0$: pre-trained BERT weights; $\boldsymbol{\theta}''_0$: randomly shuffled pre-trained UNITER weights. (†) To avoid submitting results to the VQA test server too frequently, instead of reporting results on test-dev and test-std sets, we use an internal mini-dev set for comparison. The same min-dev set was also used in UNITER. (‡) For fair comparison on transfer learning, we did not perform 2-nd stage pre-training for VCR task as used in UNITER. (⋆) To rule out other factors that may influence results besides pruning, we did not use hard neagtive mining as used in UNITER.

UNITER Can Play Lottery Tickets Too

- *Q3: Does rewinding improve performance?*

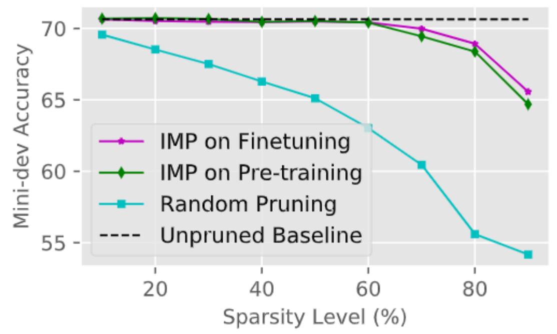


(i) VQA Rewinding

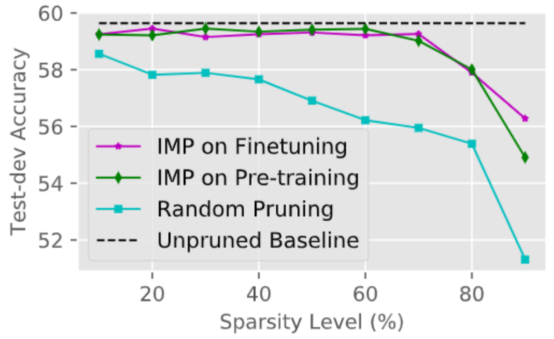
After obtaining the masks, instead of resetting the weights to θ_0, one should rewind the weights to θ_i, the weights after i steps of training

One Ticket To Win Them All

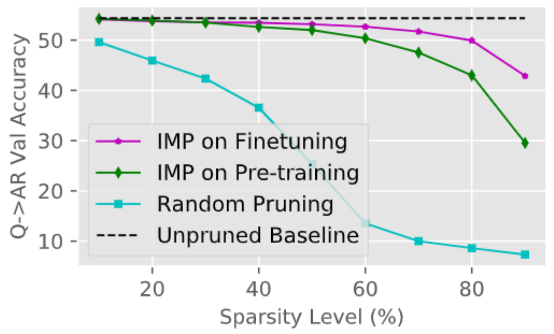
- *Q1: Do winning tickets found on pre-training tasks transfer?*



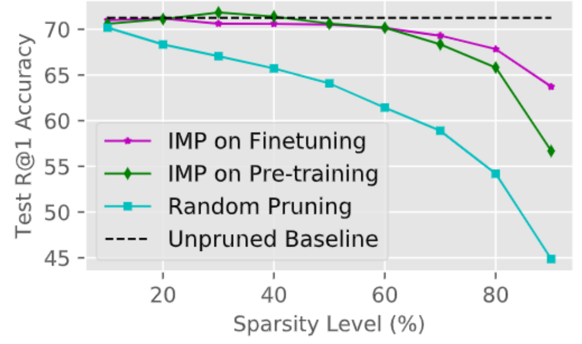
(a) VQA



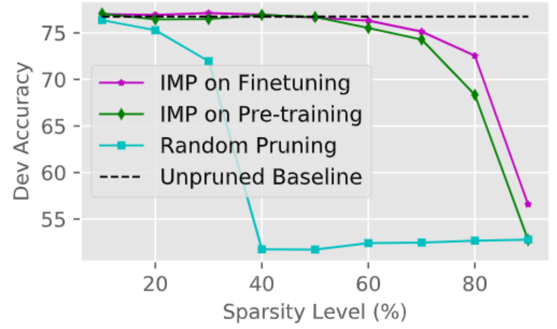
(b) GQA



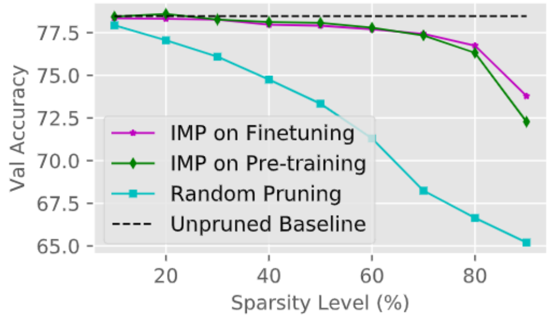
(c) VCR



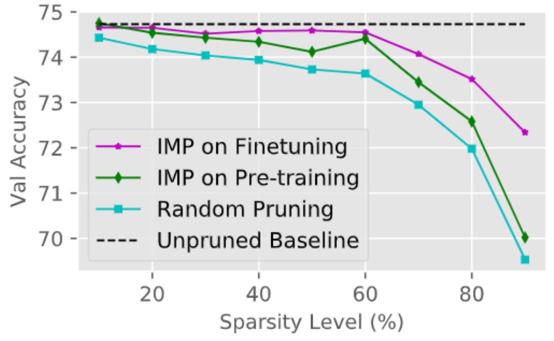
(g) Flickr30k IR



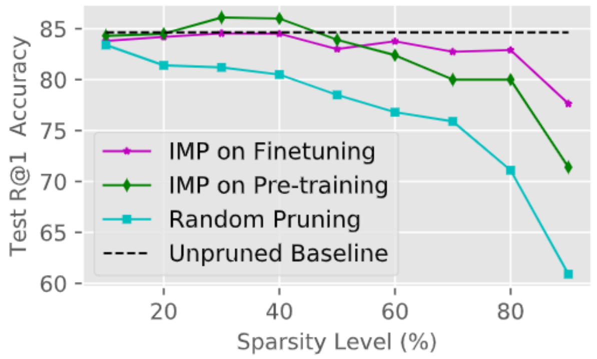
(d) NLVR^2



(e) SNLI-VE



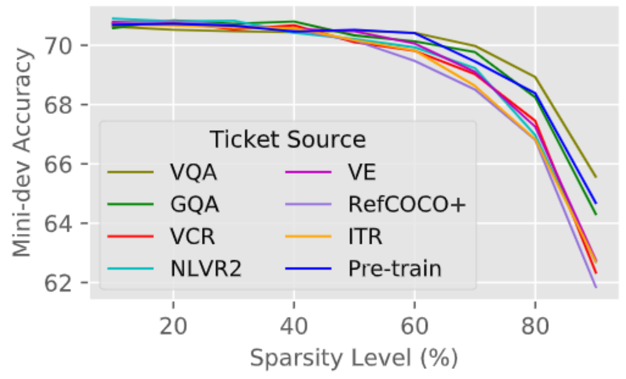
(f) RefCOCO+



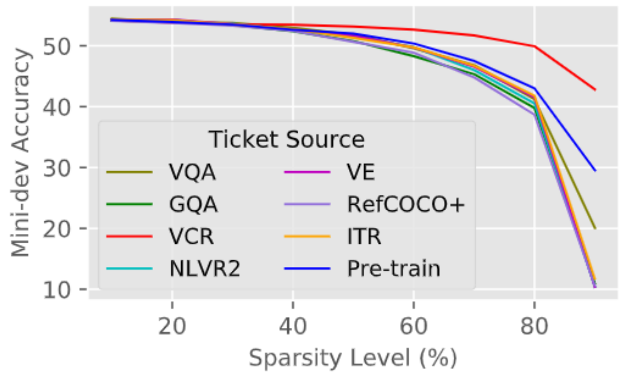
(h) Flickr30k TR

One Ticket To Win Them All

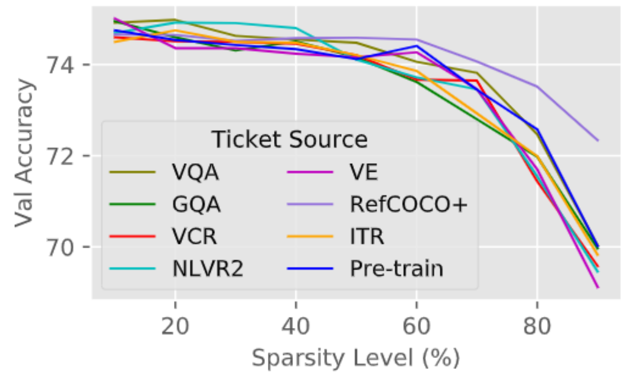
- *Q2: Do winning tickets found on downstream tasks transfer? 8*7*9 = 504 experiments*



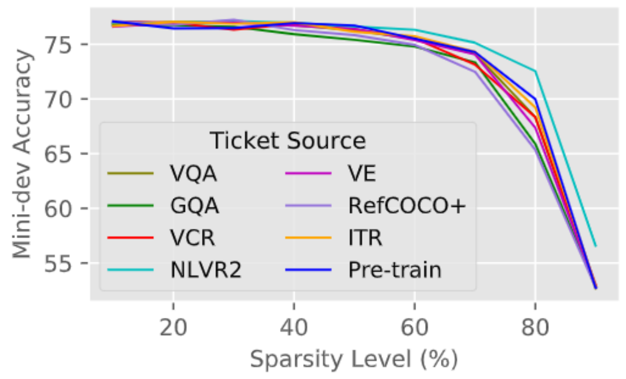
(a) VQA



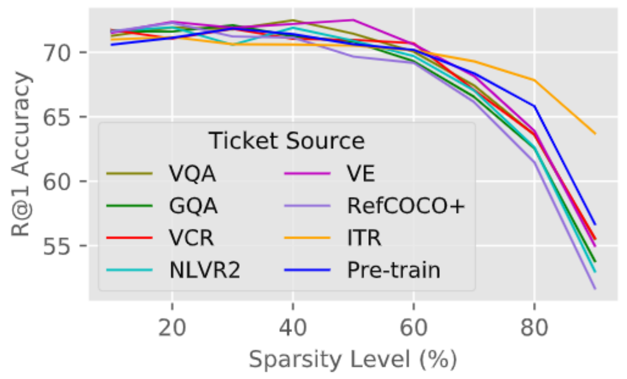
(b) VCR



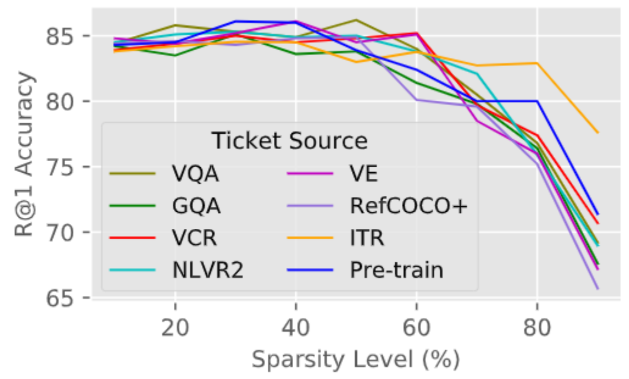
(c) RefCOCO+



(d) NLVR^2



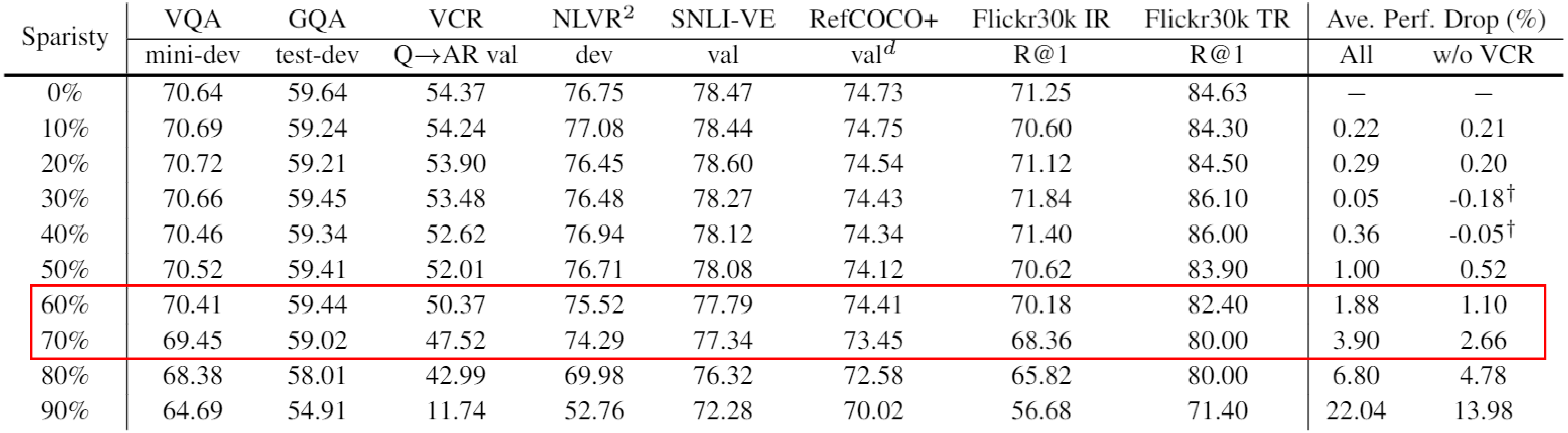
(e) Flickr30k IR



(f) Flickr30k TR

One Ticket To Win Them All

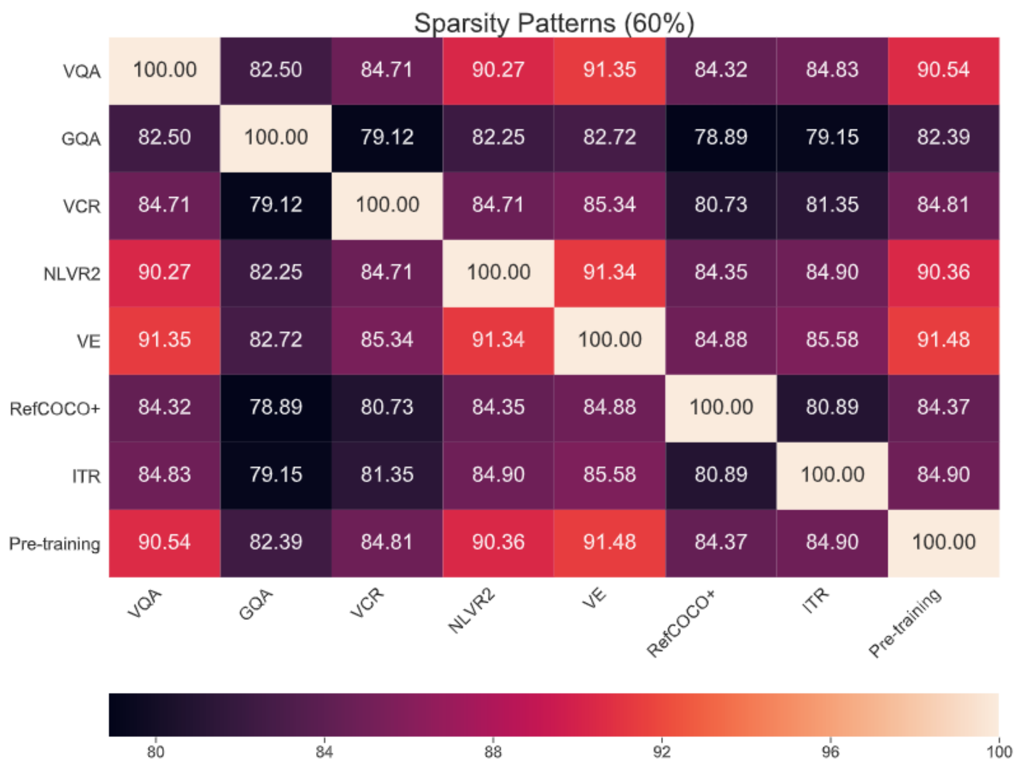
- The universal subnetwork at 60%/70% sparsity matches 98%/96% of the full accuracy over all the tasks, effectively serving as a task-agnostic compressed UNITER model.
- This number changes to 99%/97% if the VCR task is not counted in.



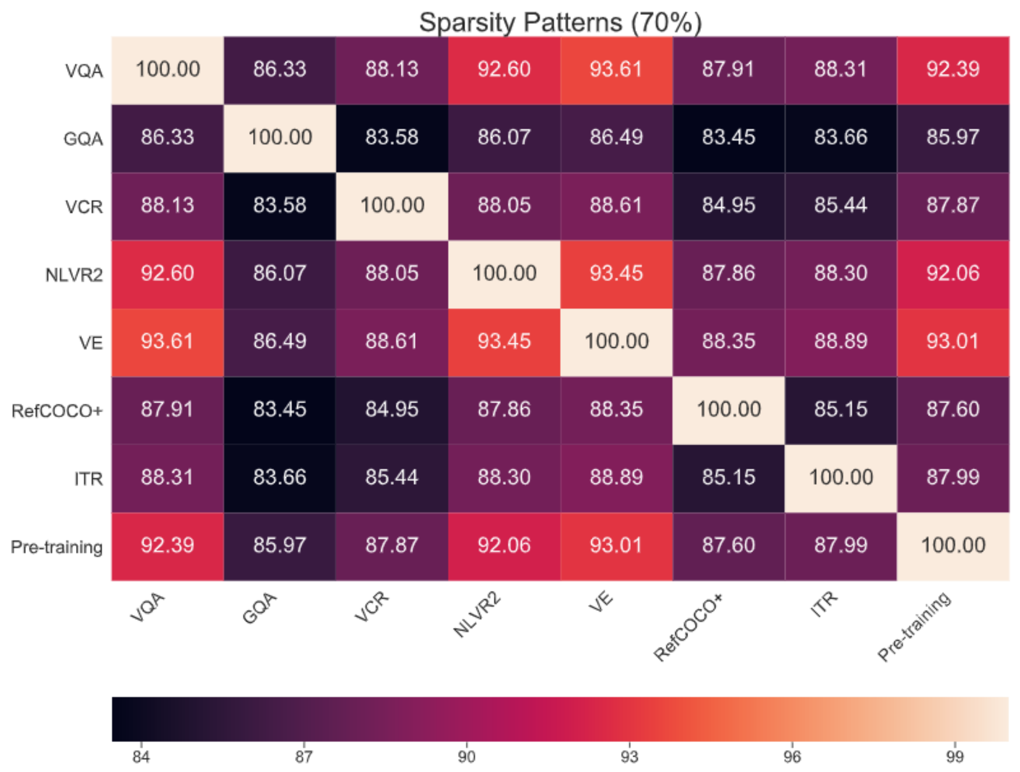
Sparisty	VQA	GQA	VCR	NLVR^2	SNLI-VE	RefCOCO+	Flickr30k IR	Flickr30k TR	Ave. Perf. Drop (%)	
	mini-dev	test-dev	Q→AR val	dev	val	val^d	R@1	R@1	All	w/o VCR
0%	70.64	59.64	54.37	76.75	78.47	74.73	71.25	84.63	–	–
10%	70.69	59.24	54.24	77.08	78.44	74.75	70.60	84.30	0.22	0.21
20%	70.72	59.21	53.90	76.45	78.60	74.54	71.12	84.50	0.29	0.20
30%	70.66	59.45	53.48	76.48	78.27	74.43	71.84	86.10	0.05	-0.18[†]
40%	70.46	59.34	52.62	76.94	78.12	74.34	71.40	86.00	0.36	-0.05[†]
50%	70.52	59.41	52.01	76.71	78.08	74.12	70.62	83.90	1.00	0.52
60%	70.41	59.44	50.37	75.52	77.79	74.41	70.18	82.40	1.88	1.10
70%	69.45	59.02	47.52	74.29	77.34	73.45	68.36	80.00	3.90	2.66
80%	68.38	58.01	42.99	69.98	76.32	72.58	65.82	80.00	6.80	4.78
90%	64.69	54.91	11.74	52.76	72.28	70.02	56.68	71.40	22.04	13.98

Intriguing Properties of the Found Masks

- Mask similarity: $\frac{m_i \bigcap m_j}{m_i \bigcup m_j}\%$ no clear patterns in the similarity of learned masks



	VQA	GQA	VCR	NLVR2	VE	RefCOCO+	ITR	Pre-training
VQA	100.00	82.50	84.71	90.27	91.35	84.32	84.83	90.54
GQA	82.50	100.00	79.12	82.25	82.72	78.89	79.15	82.39
VCR	84.71	79.12	100.00	84.71	85.34	80.73	81.35	84.81
NLVR2	90.27	82.25	84.71	100.00	91.34	84.35	84.90	90.36
VE	91.35	82.72	85.34	91.34	100.00	84.88	85.58	91.48
RefCOCO+	84.32	78.89	80.73	84.35	84.88	100.00	80.89	84.37
ITR	84.83	79.15	81.35	84.90	85.58	80.89	100.00	84.90
Pre-training	90.54	82.39	84.81	90.36	91.48	84.37	84.90	100.00

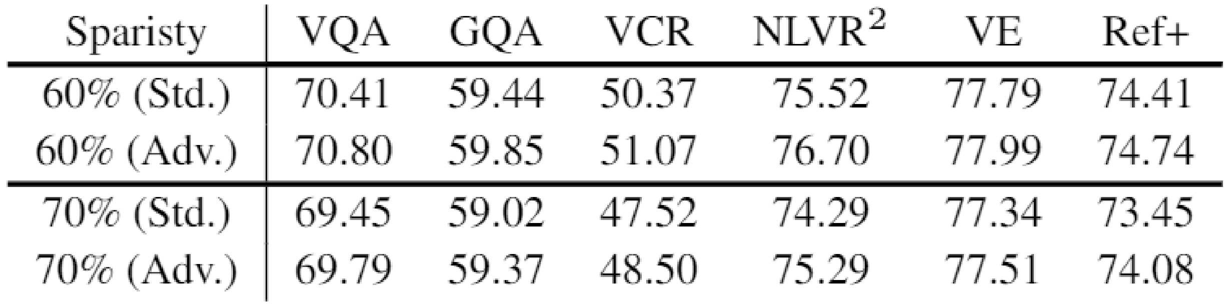


	VQA	GQA	VCR	NLVR2	VE	RefCOCO+	ITR	Pre-training
VQA	100.00	86.33	88.13	92.60	93.61	87.91	88.31	92.39
GQA	86.33	100.00	83.58	86.07	86.49	83.45	83.66	85.97
VCR	88.13	83.58	100.00	88.05	88.61	84.95	85.44	87.87
NLVR2	92.60	86.07	88.05	100.00	93.45	87.86	88.30	92.06
VE	93.61	86.49	88.61	93.45	100.00	88.35	88.89	93.01
RefCOCO+	87.91	83.45	84.95	87.86	88.35	100.00	85.15	87.60
ITR	88.31	83.66	85.44	88.30	88.89	85.15	100.00	87.99
Pre-training	92.39	85.97	87.87	92.06	93.01	87.60	87.99	100.00

Figure 1: The overlap in sparsity patterns found on each downstream task and pre-training tasks.

Lottery Tickets with Adversarial Training

- *Q1: Can AT enhance the performance of winning tickets?*

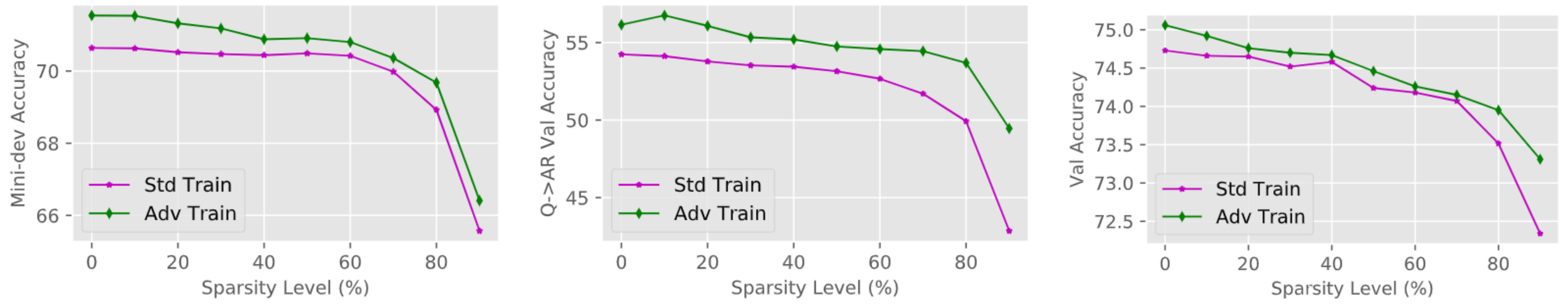


Sparisty	VQA	GQA	VCR	NLVR^2	VE	Ref+
60% (Std.)	70.41	59.44	50.37	75.52	77.79	74.41
60% (Adv.)	70.80	59.85	51.07	76.70	77.99	74.74
70% (Std.)	69.45	59.02	47.52	74.29	77.34	73.45
70% (Adv.)	69.79	59.37	48.50	75.29	77.51	74.08

Table 3: Performance of adversarial training on the universal sub-networks at 60% and 70% sparsities. Std.: standard cross-entropy training; Adv.: adversarial training. Ref+: RefCOCO+.

Lottery Tickets with Adversarial Training

- *Q2: Can we also use AT to find winning tickets?*



(a) VQA (b) VCR (c) RefCOCO+

Figure 4: Performance of subnetworks that are found by adversarial training on the tasks of VQA, VCR and RefCOCO+.

Limitations

- *Efficiency:* We mainly focused on the scientific study of LTH. For future work, we plan to investigate the real speedup results on a hardware platform that is friendly to unstructured pruning (XNNPACK).
- *Object Detection*: Most VL research focused on multi-modal fusion, while image features are often extracted by an offline object detection model. Therefore, we mainly studied the LTH of the transformer module. For future work, we are also interested in studying the LTH of object detection and the combination of both.
- *Generalization*: We mainly focused on UNITER, but we believe our findings will also hold for other VLP models, as they all essentially share the same transformer structure.

Playing Lottery Tickets with Vision and Language

- Contributions:
  - The first empirical study on lottery ticket hypothesis for vision and language
  - Use adversarial training to enhance pruning performance
  - A released VLP model ➔ *A "lucky" UNITER* that draws the lottery and survives from pruning
- Moving forward:
  - *Early-bird lottery tickets*: identify structured sparsity patterns early in the training, rather than repeating the train-prune-retrain cycle with unstructured pruning for real speedup
  - *Dynamic sparse training*: grow and prune subnetworks on the fly

UNITER