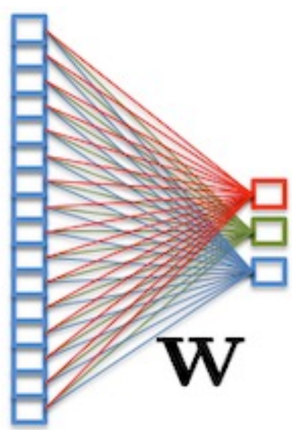
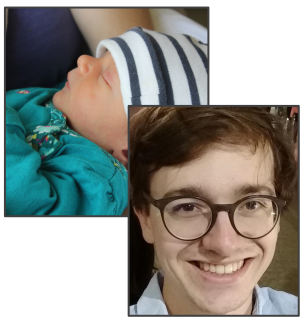
5IF - Deep Learning and Differentiable Programming

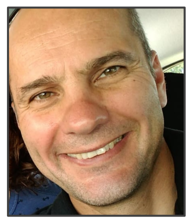
6.2 One concrete application of theory in DL



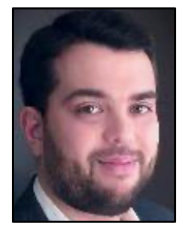
INSA LYON Christian Wolf

Supervising the Transfer of Reasoning Patterns in VQA


Corentin Kervadec *


Christian Wolf *


Grigory Antipov


Moez Baccouche

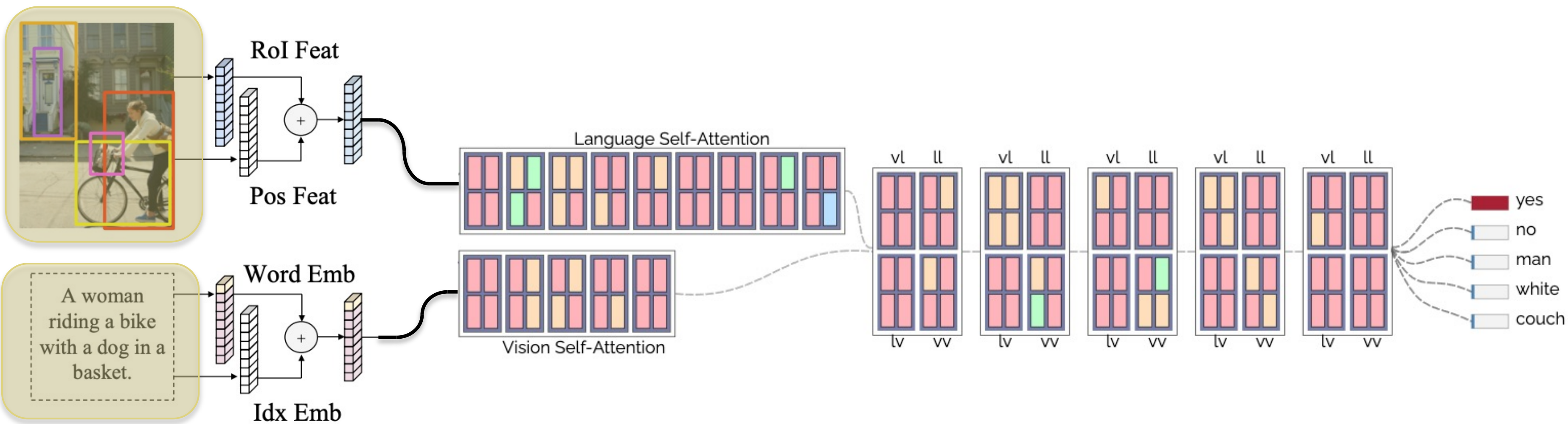

Madiha Nadri

Visual Question Answering

Is the cabbage to the left of the broccoli?

Transformer based models

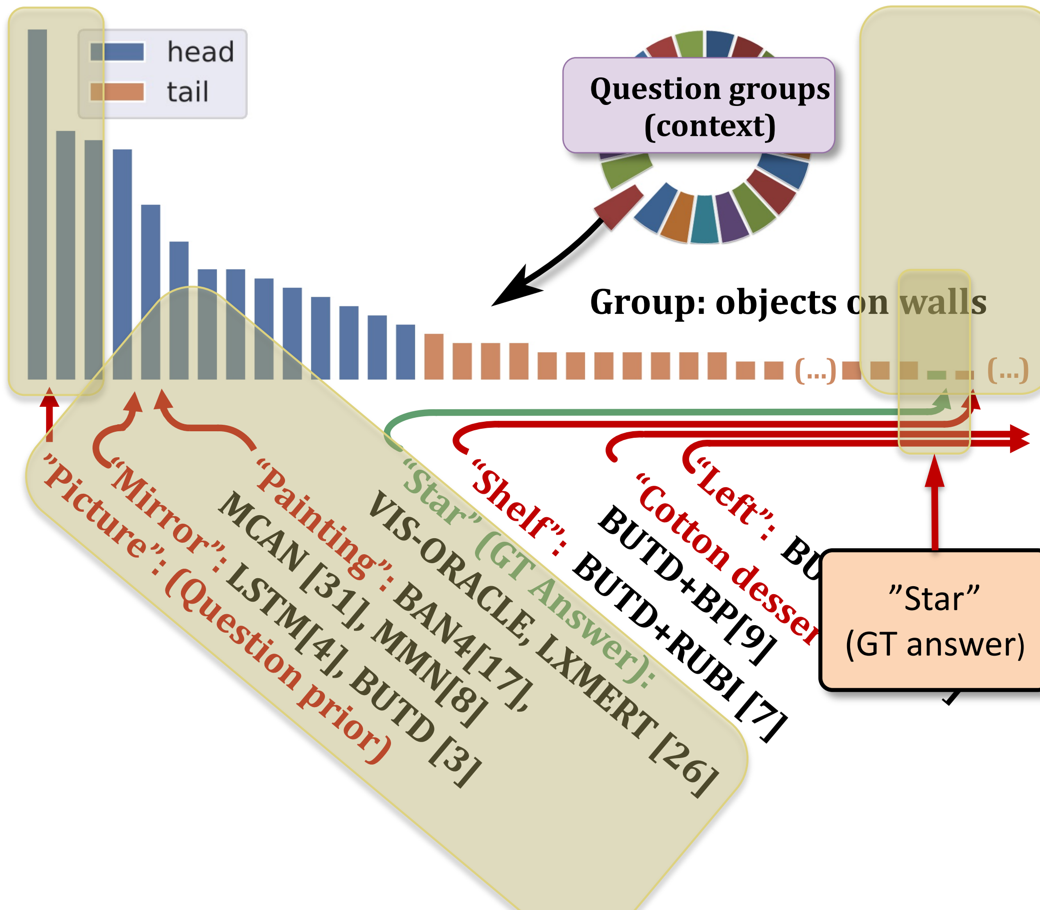
A vision and language encoder with self-attention and cross-attention.



Tan, H. and Bansal, M. LXMERT: Learning cross-modality encoder representations from transformers. EMNLP-IJCNLP 2019.

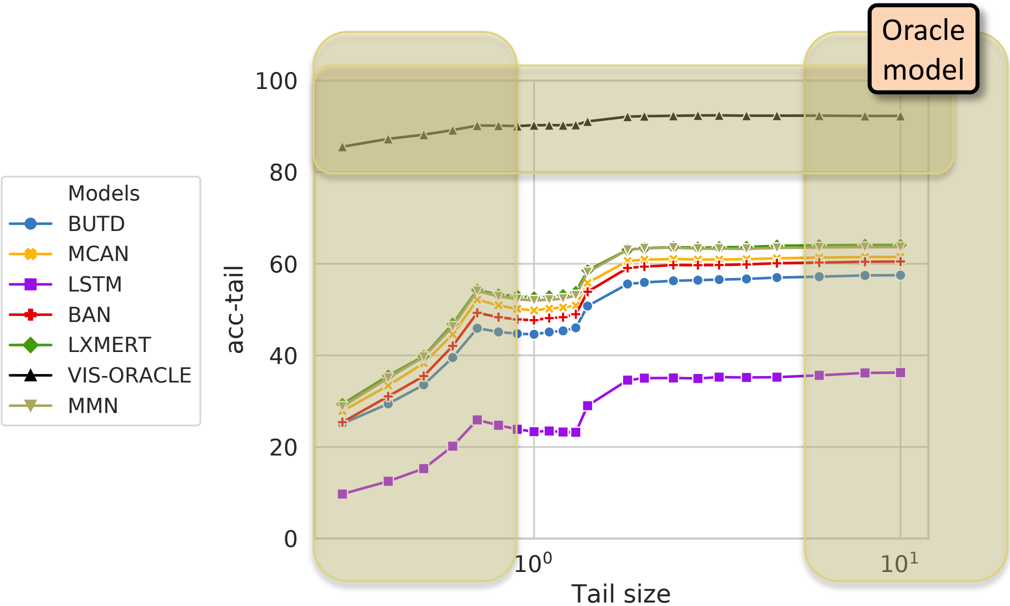
Reasoning vs. bias exploitation (1)

"What is on the wall?"



C. Kervadec, G. Antipov, M. Baccouche and C. Wolf, Roses Are Red, Violets Are Blue... but Should VQA Expect Them To? CVPR 2021.

Reasoning vs. bias exploitation (2)

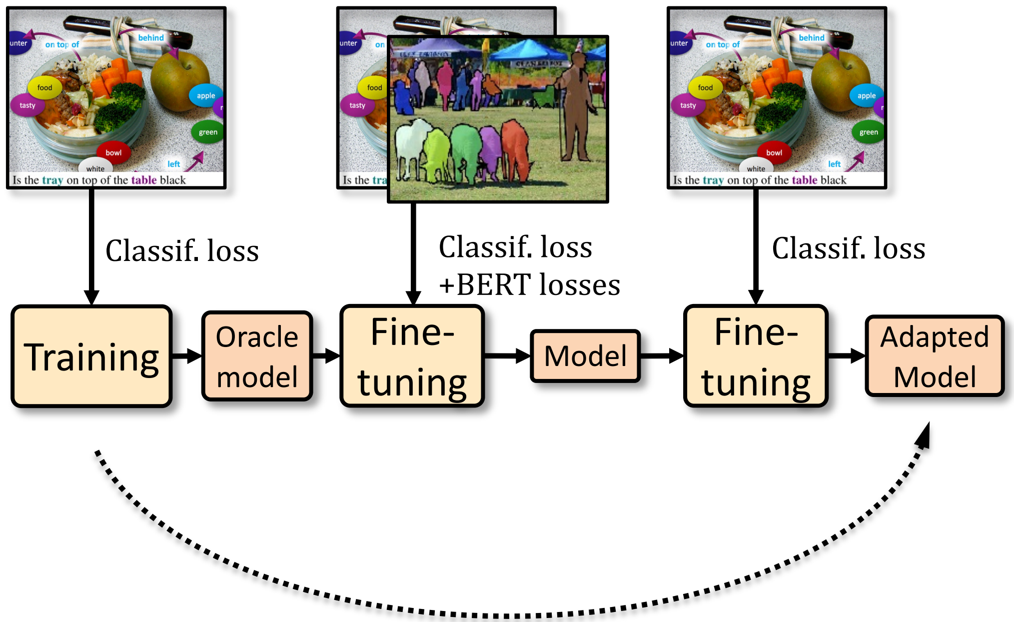


C. Kervadec, T. Jaunet, G. Antipov, M. Baccouche, R. Vuillemot and C. Wolf, How Transferrable are Reasoning Patterns in VQA? CVPR 2021.

XAI – explainable AI

T. Jaunet, C. Kervadec, G. Antipov, M. Baccouche, R. Vuillemot and C. Wolf. ***VisQA: X-raying Vision and Language Reasoning*** *in Transformers. IEEE-T. on Visualization and Computer Graphics 2021.*

Oracle transfer

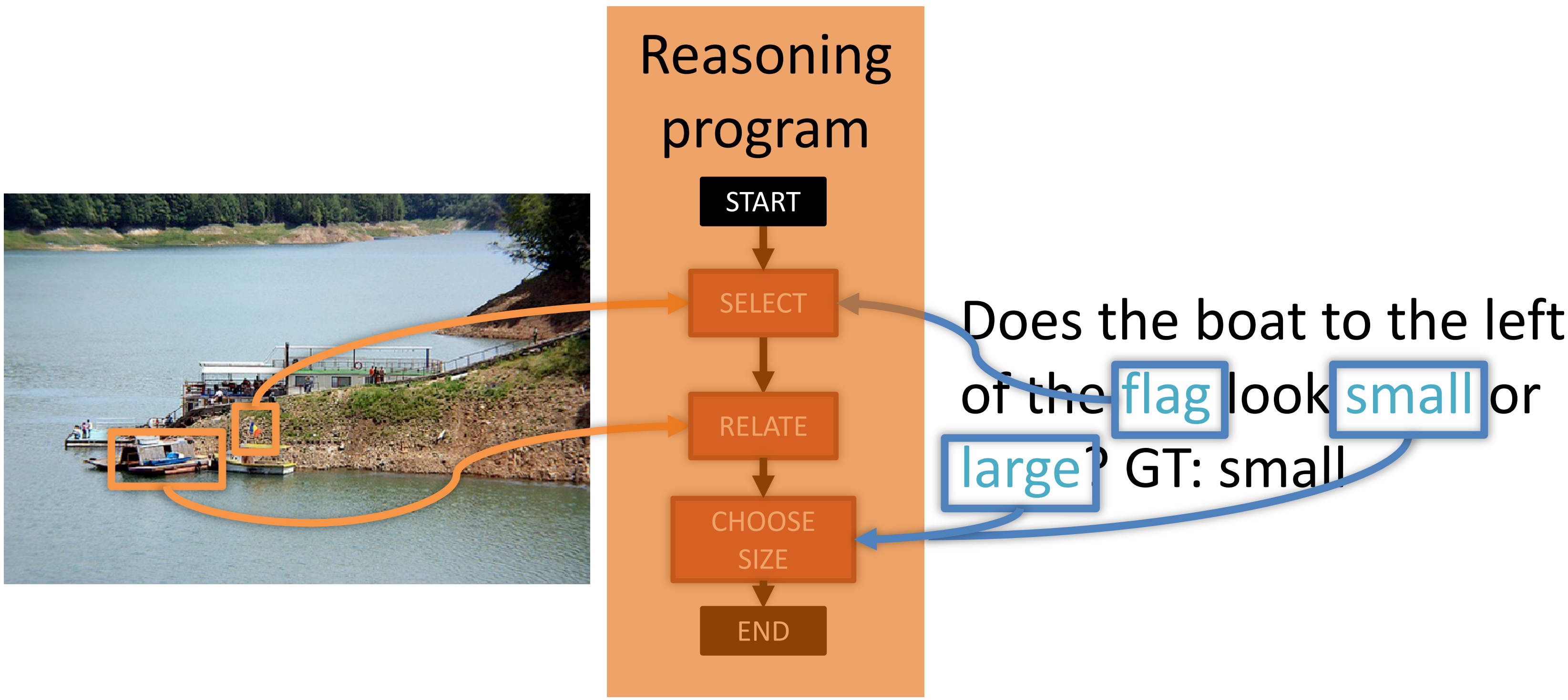


Motivation of this work (NeurIPS)

Loss of reasoning capacity after transfer to noisy input

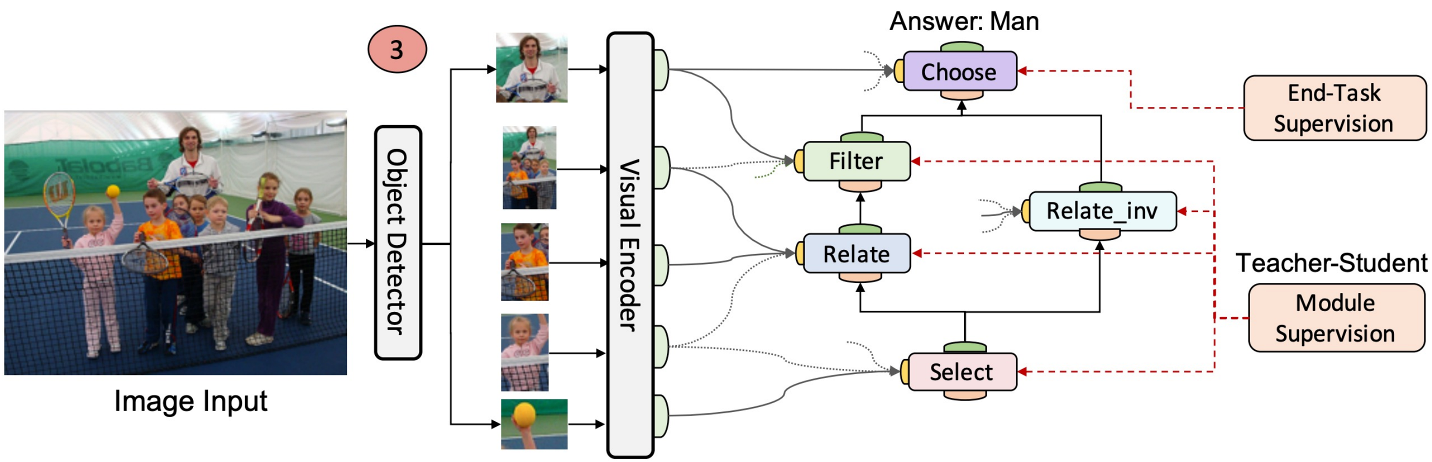
C. Kervadec, T. Jaunet, G. Antipov, M. Baccouche, R. Vuillemot and C. Wolf, How Transferrable are Reasoning Patterns in VQA? CVPR 2021.

Ground truth reasoning programs



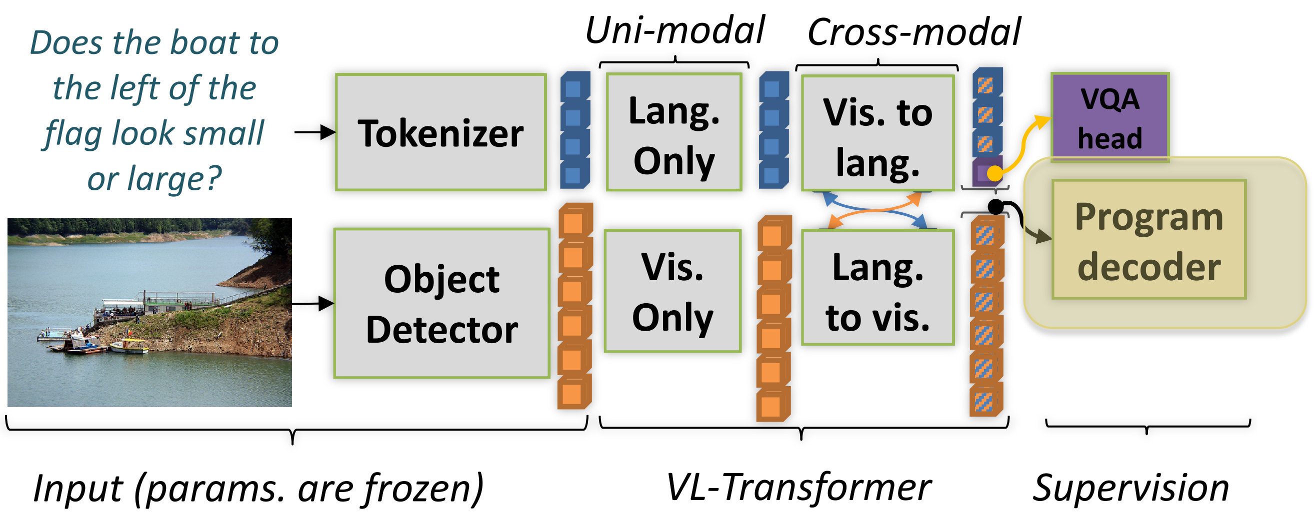
D.A. Hudson and C.D. Manning. GQA: A new dataset for real-world visual reasoning and compositional question answering, CVPR 2019.

Neuro-symbolic reasoning



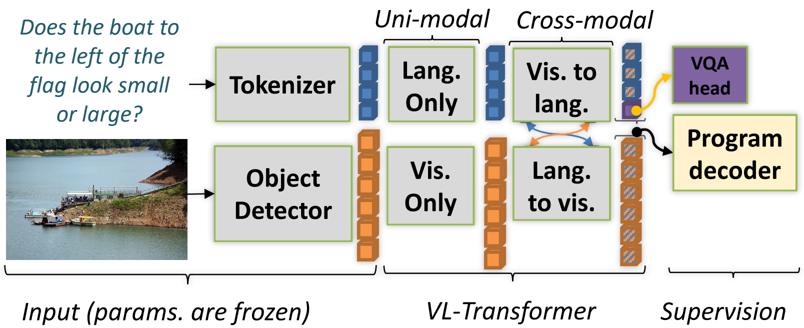
W. Chen, Z. Gan, L. Li, Y. Cheng, W. Wang, and J. Liu. Meta module network for compositional visual reasoning. WACV, 2021

Supervising program prediction

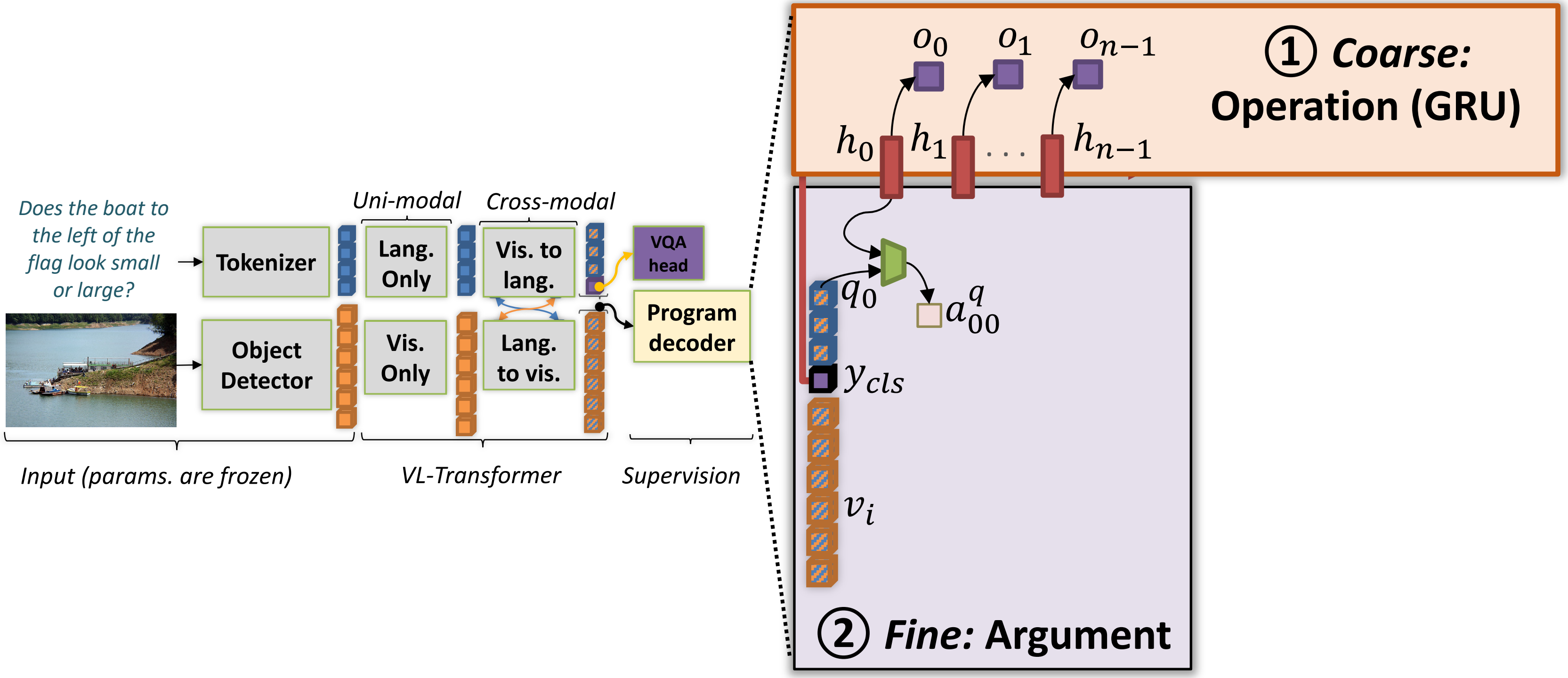


Tan, H. and Bansal, M. LXMERT: Learning cross-modality encoder representations fromtransformers. EMNLP-IJCNLP 2019.

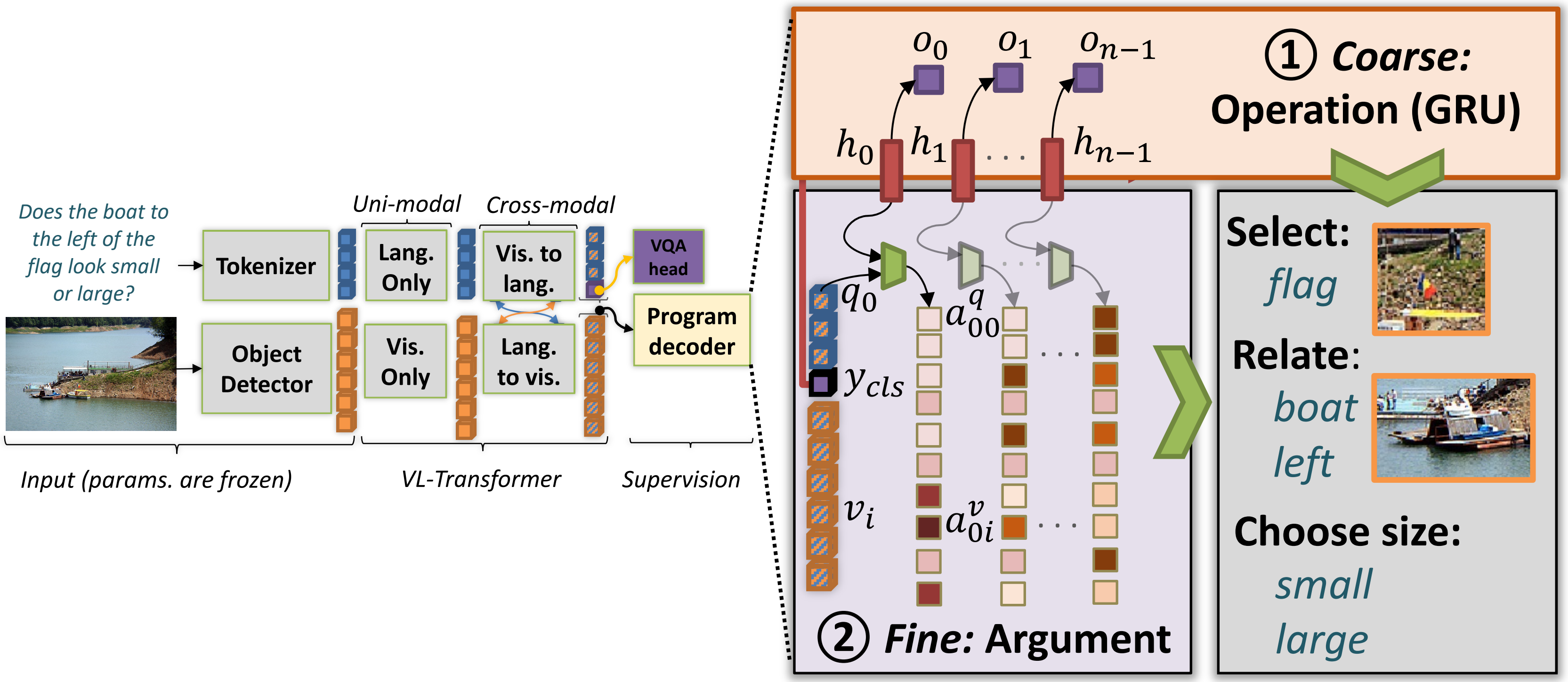
Supervising program prediction



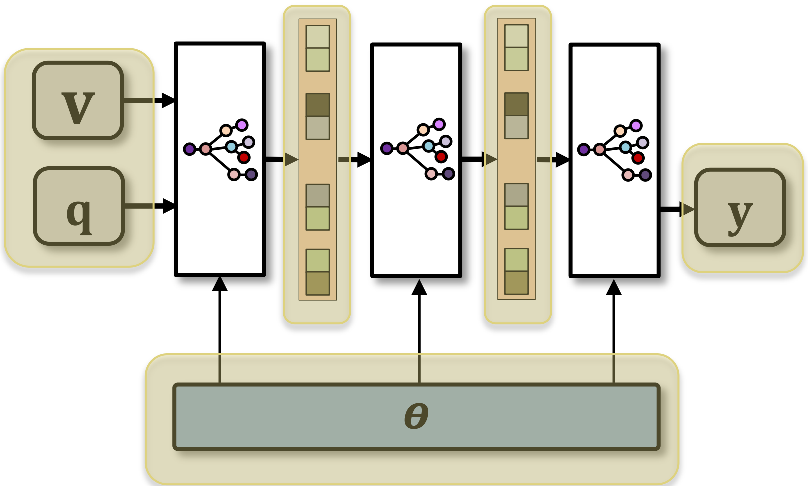
Supervising program prediction

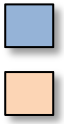


Supervising program prediction



Sample complexity

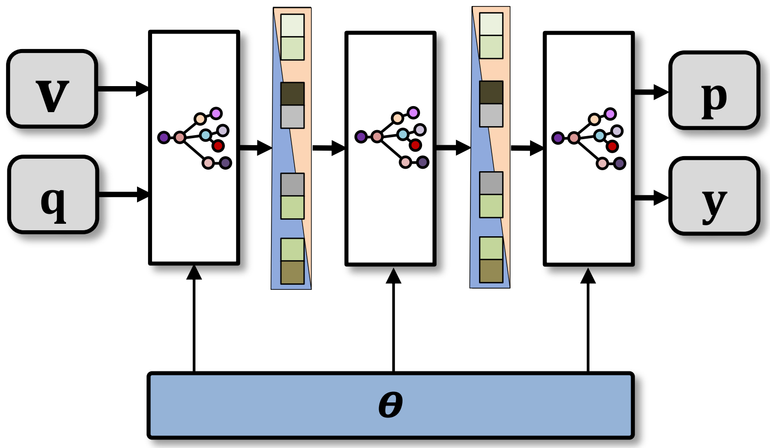




Learned knowledge of the reasoning processes

Latent variables necessary for reasoning over multiple hops

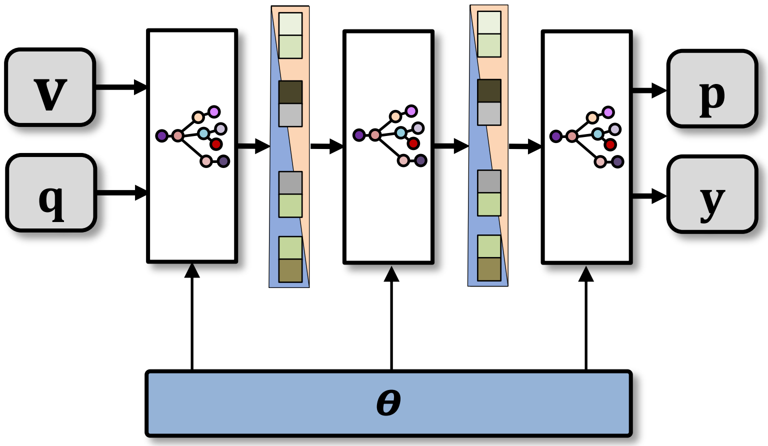
Sample complexity



Learned knowledge of the reasoning processes

Latent variables necessary for reasoning over multiple hops

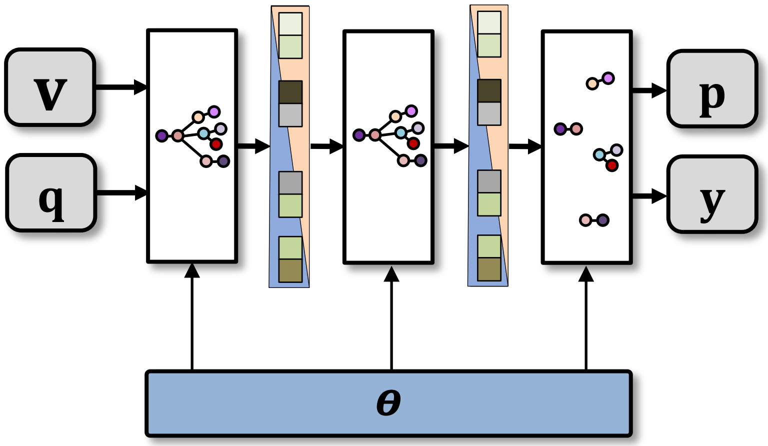
Sample complexity



Learned knowledge of the reasoning processes

Latent variables necessary for reasoning over multiple hops

Sample complexity



Learned knowledge of the reasoning processes

Latent variables necessary for reasoning over multiple hops

Sample complexity of overparam. MLPs

Theorem [1][2]: Let $\mathcal{A}$ be an overparametrized and randomly initialized two-layer MLP trained with gradient descent for a sufficient number of iterations. Suppose $g : \mathbb{R}^d \to \mathbb{R}^m$ with components $g(x)^{(i)} = \sum_j \alpha_j^{(i)} (\beta_j^{(i)T} x)^{p_j^{(i)}}$, where $\beta_j^{(i)} \in \mathbb{R}^d$, $\alpha^{(i)} \in \mathbb{R}$, and $p_j^{(i)} = 1$ or $p_j^{(i)} = 2l, l \in \mathbb{N}_+$. The sample complexity $\mathcal{C}_{\mathcal{A}}(g, \epsilon, \delta)$ is

$$\mathcal{C}_{\mathcal{A}}(g, \epsilon, \delta) = O\left(\frac{\max_i \sum_j p_j^{(i)} |\alpha_j^{(i)}| \cdot ||\beta_j^{(i)}||_2^{p_j^{(i)}} + \log(\frac{m}{\delta})}{(\epsilon/m)^2} \right)$$

[1] S.S. Du S. Arora, W. Hu, Z. Li, and R. Wang. Fine-grained Analysis of optimization and generalization for overparametrized two-layer neural networks. ICML, 2019.

[2] K. Xu, J. Li, M. Zhang, S.S. Du, K.-I. K., and S. Jegelka. What can Neural Networks Reason About. ICLR, 2020.

Assumption: decomposition of reasoning

The unknown reasoning function

$$\boldsymbol{y}^* = g(\boldsymbol{q}, \boldsymbol{v})$$

Assumption: decomposition of reasoning

The unknown reasoning function

$$\boldsymbol{y}^* = g(\boldsymbol{q}, \boldsymbol{v})$$

is a mixture model, which decomposes:

$$\boldsymbol{y}^* = \sum_r \boxed{\boldsymbol{\pi}_r(\boldsymbol{q})}\boxed{g_r(\boldsymbol{v})},$$

Assumption: decomposition of reasoning

The unknown reasoning function

$$\boldsymbol{y}^* = g(\boldsymbol{q}, \boldsymbol{v})$$

is a mixture model, which decomposes:

$$\boldsymbol{y}^* = \sum_r \boldsymbol{\pi}_r(\boldsymbol{q}) g_r(\boldsymbol{v}),$$

where each reasoning mode is a multi-variate polynomial:

$$\boldsymbol{h}_r^{(i)} = g_r(\boldsymbol{v}) = \sum_j \alpha_{r,j}^{(i)} (\beta_{r,j}^{(i)T} \boldsymbol{v})^{p_{r,j}^{(i)}}$$

with parameters $\omega = \left\{ \alpha_{r,j}^{(i)}, \beta_{r,j}^{(i)}, p_{r,j}^{(i)} \right\}$.

Assumption: mode selector

$$\boldsymbol{\pi}_r(\boldsymbol{q}) \quad ?$$

Assumption: mode selector

Assumption The input question embeddings $\boldsymbol{q}$ are separated into clusters according to reasoning modes r, such that the underlying reasoning mode estimator g_π can be realized as a NN classifier with dot-product similarity in this embedding space.

Assumption: mode selector

Assumption The input question embeddings $\boldsymbol{q}$ are separated into clusters according to reasoning modes r, such that the underlying reasoning mode estimator g_π can be realized as a NN classifier with dot-product similarity in this embedding space.

$\longrightarrow$ the reasoning mode estimator can be expressed as a generalized linear model:

$$\boldsymbol{\pi} = g_\pi(\boldsymbol{q}) = \sigma\left(\left[\gamma_0^T \boldsymbol{q},\ \gamma_1^T \boldsymbol{q}, \ldots\right]\right), \tag{1}$$

The full reasoning function

$$\boldsymbol{y}^{*(i)} = \sum_r \sum_j (\gamma_r^T \boldsymbol{x}) \alpha_{r,j}^{(i)} (\beta_{r,j}^{(i)T} \boldsymbol{x})^{p_{r,j}^{(i)}}$$

The full reasoning function

$$\boldsymbol{y}^{*(i)} = \sum_r \sum_j (\gamma_r^T \boldsymbol{x}) \alpha_{r,j}^{(i)} (\beta_{r,j}^{(i)T} \boldsymbol{x})^{p_{r,j}^{(i)}}$$

Theorem: Let $\mathcal{A}$ be an overparametrized and randomly initialized two-layer MLP trained with gradient descent for a sufficient number of iterations. Suppose $g : \mathbb{R}^d \to \mathbb{R}^m$ with components $g(x)^{(i)} = \sum_r \sum_j (\gamma_r^T \boldsymbol{x}) \alpha_{r,j}^{(i)} (\beta_{r,j}^{(i)T} \boldsymbol{x})^{p_{r,j}^{(i)}}$ where $\gamma_r \in \mathbb{R}^d$, $\beta_{r,j}^{(i)} \in \mathbb{R}^d$, $\alpha_{r,j}^{(i)} \in \mathbb{R}$, and $p_{r,j}^{(i)} = 1$ or $p_{r,j}^{(i)} = 2l, l \in \mathbb{N}_+$. The sample complexity $\mathcal{C}_{\mathcal{A}}(g, \epsilon, \delta)$ is

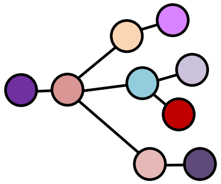
$$\mathcal{C}_{\mathcal{A}}(g, \epsilon, \delta) = O\left(\frac{\max_i \sum_r \sum_j \pi p_{r,j}^{(i)} |\alpha| \cdot ||\gamma_r||_2 \cdot ||\beta_{r,j}||_2^{p_{r,j}^{(i)}} + \log(m/\delta)}{(\epsilon/m)^2} \right).$$

(Proof in the paper)

Complexity with program supervision

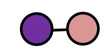
Assumption: through supervising reasoning programs, learning is separated into several different processes,

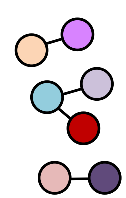
1. learning of the reasoning mode estimator $g_\pi()$;
2. learning of the the different reasoning modules learned independently.



Complexity with program supervision

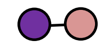
Assumption: through supervising reasoning programs, learning is separated into several different processes,

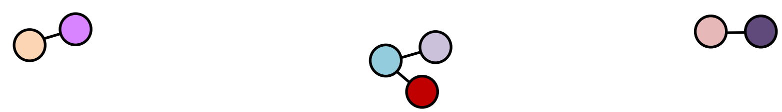
1. learning of the reasoning mode estimator $g_\pi()$; 
2. learning of the the different reasoning modules learned independently.



Complexity with program supervision

Assumption: through supervising reasoning programs, learning is separated into several different processes,

1. learning of the reasoning mode estimator $g_\pi()$; 
2. learning of the the different reasoning modules learned independently.



Difference in complexity

Under very <u>conservative</u> assumptions, the gain in complexity is

$$\sqrt{2}\frac{\Gamma(\frac{m}{2}+\frac{1}{2})}{\Gamma(\frac{m}{2})}$$

(Proof in the paper)

Impact of program supervision

	Model	Oracle transf.	Prog. sup.	GQA-OOD acc-tail	GQA-OOD acc-head	GQA test-dev	GQA binary*	GQA open*	GQA test-std	AUC† prog.
scratch	(a) Baseline			42.9	49.5	52.4	-	-	-	/
scratch	(b) Oracle transfer	✓		48.2$\pm$0.3	54.6$\pm$1.1	57.0$\pm$0.3	74.5	42.1	57.3	/
scratch	**(c) Ours**	✓	✓	**48.8**$\pm$0.1	**56.1**$\pm$0.3	**57.8**$\pm$0.2	**75.4**	**43.0**	**58.2**	97.1
+ lxmert	(d) Baseline			47.5	55.2	58.5	-	-	-	/
+ lxmert	(e) Oracle transfer	✓		47.1	54.8	58.4	77.1	42.6	58.8	/
+ lxmert	(f) **Ours**	✓	✓	**48.0**$\pm$0.6	**56.6**$\pm$0.6	**59.3**$\pm$0.3	**77.3**	**44.1**	**59.7**	96.4

BERT/LXMERT pre-training on GQA unbalanced training.
Scores on GQA (*test-dev* and *test-std*) and GQA-OOD (*test*). * binary and open scores are computed on the test-std.

† we evaluate visual argument prediction by computing AUC0.66 on GQA-val.

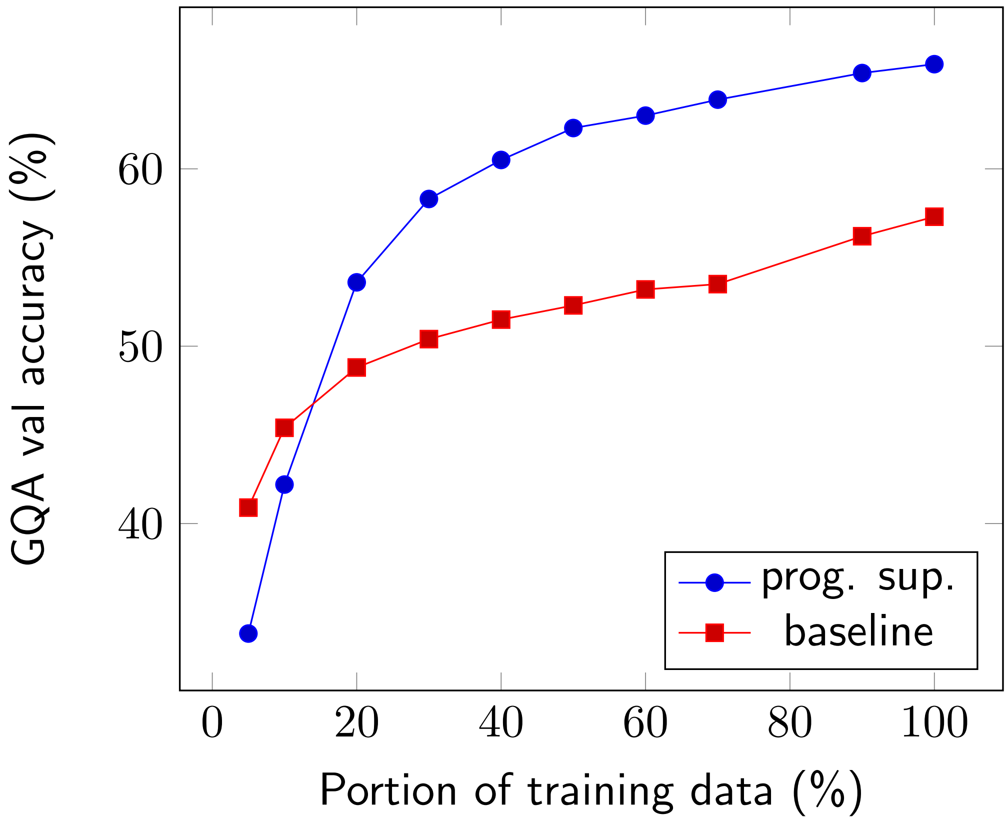
Impact of types of supervision

Ablations	GQA-OOD acc-tail (val.)	GQA val.
(1) VQA only	46.9	62.2
(2) Coarse only	46.5	62.5
(3) Coarse + dep.	46.8	62.8
(4) Full w/o v.arg	47.3	63.7
(5) Full (ours)	**49.9**	**66.2**

Compact model, no LXMERT/BERT pre-training, no Oracle
GQA validation set

v.arg = supervision of visual arguments

Empirical estimation of sample complexity



Comparison with SOTA

Method	Visual feats.	Additional supervision	Training data (M) Img	Sent	GQA-OOD acc-tail	acc-head	GQA bin.	open	all
BAN4	RCNN	-	$\approx$ 0.1	$\approx$1	47.2	51.9	76.0	40.4	57.1
MCAN	RCNN	-	$\approx$ 0.1	$\approx$1	46.5	53.4	75.9	42.2	58.0
Oracle transfer	RCNN	-	$\approx$0.18	$\approx$1	48.3	55.5	75.2	44.1	58.7
MMN	RCNN	Program	$\approx$0.1	$\approx$15	48.0	55.5	78.9	44.9	60.8
LXMERT	RCNN	-	$\approx$0.18	$\approx$9	**49.8**	57.7	77.8	45.0	60.3
Ours	VinVL	Program	$\approx$0.1	$\approx$15	49.1	**59.7**	80.1	48.0	63.0
NSM	SG	Scene graph	$\approx$0.1	$\approx$1	-	-	78.9	**49.3**	63.2
OSCAR+VinVL	VinVL	-	$\approx$5.7	$\approx$9	-	-	**82.3**	48.8	**64.7**

Conclusion

- We exploit the fact that Oracle models are less prone to shortcut learning.
- We train an Oracle model and transfer to a deployable model.
- We supervise program prediction during the transfer to maintain a strong link to the objective.
- We theorectially show that program supervision decreases sample complexity under some assumptions.