

Compositional Generalization



Soham Dan, Parikshit Ram
IBM Research

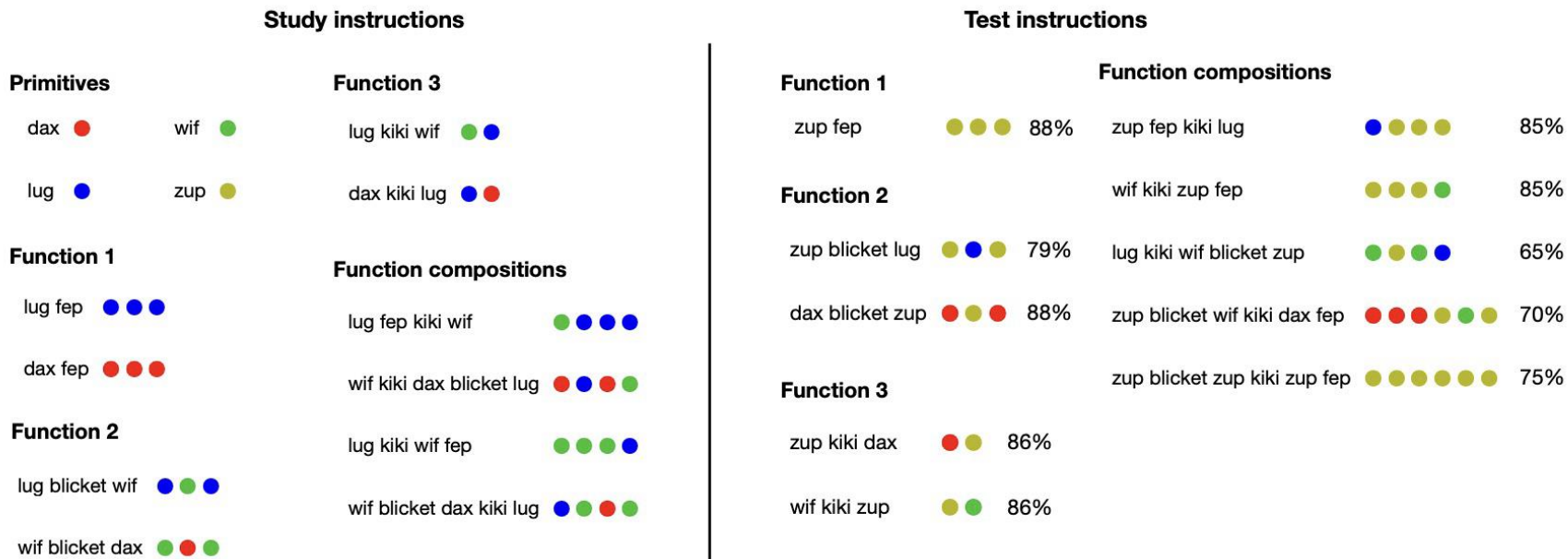
Course Plan

- What is Compositional Generalization?
- Compositional Generalization Benchmarks
- Approaches to Compositional Generalization



Human Few-Shot Learning of Compositional Instructions

(Lake et al, 2019)



While humans are able to perform well on all the compositional tasks, RNNs get only 2.5% correct on the test instructions.

Indicates humans utilize inductive biases that are lacking in neural networks

Defining Compositionality

(Fodor & Pylyshyn, 1988; Partee, 1995; Pagin & Westerstahl, 2010; Hupkes et al., 2020)

The meaning of the whole is a function of the meaning of the parts and of the way they are syntactically combined.

Systematicity: Can we recombine known parts/rules with novel concepts?

walk thrice, dax twice → **dax thrice, walk twice**

Productivity: Can we generalize to longer/shorter sentences ?

walk, walk after walk → **walk after walk after walk**

Substitutivity: Can we generalize to meaning-preserving substitutions ?

walk thrice → **step thrice**

Benchmarks for Testing Compositional Generalization

Modern Neural Networks get very **high accuracies** in the **IID test split** (interpolation setting).

Several benchmarks have been proposed to measure their performance in **compositionally structured OOD test splits** (extrapolation setting).

Course Plan

- What is Compositional Generalization?
- Compositional Generalization Benchmarks
- Approaches to Compositional Generalization



SCAN: Simplified Version of CommonAI Navigation Tasks

(Lake & Baroni, 2018)

Task: Natural language translation to navigation action commands

Ground-truth: (Bounded) Context-Free Grammar for source language, and target language interpretations for CFG terminals

Forms of generalization:

- Random split: Standard, IID setting where the entire dataset is randomly split 80/20 into train/test
- Add-jump split: Novel composition of known primitives
- Length split: Longer target sequences than in training
- MCD split: Similar atom distributions, different compound distributions

SCAN Examples

jump	⇒	JUMP
jump left	⇒	LTURN JUMP
jump around right	⇒	RTURN JUMP RTURN JUMP RTURN JUMP RTURN JUMP
turn left twice	⇒	LTURN LTURN
jump thrice	⇒	JUMP JUMP JUMP
jump opposite left and walk thrice	⇒	LTURN LTURN JUMP WALK WALK WALK
jump opposite left after walk around left	⇒	LTURN WALK LTURN WALK LTURN WALK LTURN WALK LTURN LTURN JUMP

SCAN Grammar & Interpretation

$C \rightarrow S \text{ and } S$	$V \rightarrow D[1] \text{ opposite } D[2]$	$D \rightarrow \text{turn left}$
$C \rightarrow S \text{ after } S$	$V \rightarrow D[1] \text{ around } D[2]$	$D \rightarrow \text{turn right}$
$C \rightarrow S$	$V \rightarrow D$	$U \rightarrow \text{walk}$
$S \rightarrow V \text{ twice}$	$V \rightarrow U$	$U \rightarrow \text{look}$
$S \rightarrow V \text{ thrice}$	$D \rightarrow U \text{ left}$	$U \rightarrow \text{run}$
$S \rightarrow V$	$D \rightarrow U \text{ right}$	$U \rightarrow \text{jump}$

$\llbracket \text{walk} \rrbracket = \text{WALK}$

$\llbracket \text{look} \rrbracket = \text{LOOK}$

$\llbracket \text{run} \rrbracket = \text{RUN}$

$\llbracket \text{jump} \rrbracket = \text{JUMP}$

$\llbracket \text{turn left} \rrbracket = \text{LTURN}$

$\llbracket \text{turn right} \rrbracket = \text{RTURN}$

$\llbracket u \text{ left} \rrbracket = \text{LTURN } \llbracket u \rrbracket$

$\llbracket u \text{ right} \rrbracket = \text{RTURN } \llbracket u \rrbracket$

$\llbracket \text{turn opposite left} \rrbracket = \text{LTURN LTURN}$

$\llbracket \text{turn opposite right} \rrbracket = \text{RTURN RTURN}$

$\llbracket u \text{ opposite left} \rrbracket = \llbracket \text{turn opposite left} \rrbracket \llbracket u \rrbracket$

$\llbracket u \text{ opposite right} \rrbracket = \llbracket \text{turn opposite right} \rrbracket \llbracket u \rrbracket$

$\llbracket \text{turn around left} \rrbracket = \text{LTURN LTURN LTURN LTURN}$

$\llbracket \text{turn around right} \rrbracket = \text{RTURN RTURN RTURN RTURN}$

$\llbracket u \text{ around left} \rrbracket = \text{LTURN } \llbracket u \rrbracket \text{ LTURN } \llbracket u \rrbracket \text{ LTURN } \llbracket u \rrbracket \text{ LTURN } \llbracket u \rrbracket$

$\llbracket u \text{ around right} \rrbracket = \text{RTURN } \llbracket u \rrbracket \text{ RTURN } \llbracket u \rrbracket \text{ RTURN } \llbracket u \rrbracket \text{ RTURN } \llbracket u \rrbracket$

$\llbracket x \text{ twice} \rrbracket = \llbracket x \rrbracket \llbracket x \rrbracket$

$\llbracket x \text{ thrice} \rrbracket = \llbracket x \rrbracket \llbracket x \rrbracket \llbracket x \rrbracket$

$\llbracket x_1 \text{ and } x_2 \rrbracket = \llbracket x_1 \rrbracket \llbracket x_2 \rrbracket$

$\llbracket x_1 \text{ after } x_2 \rrbracket = \llbracket x_2 \rrbracket \llbracket x_1 \rrbracket$

SCAN Experimental Results (subset): Add-jump split

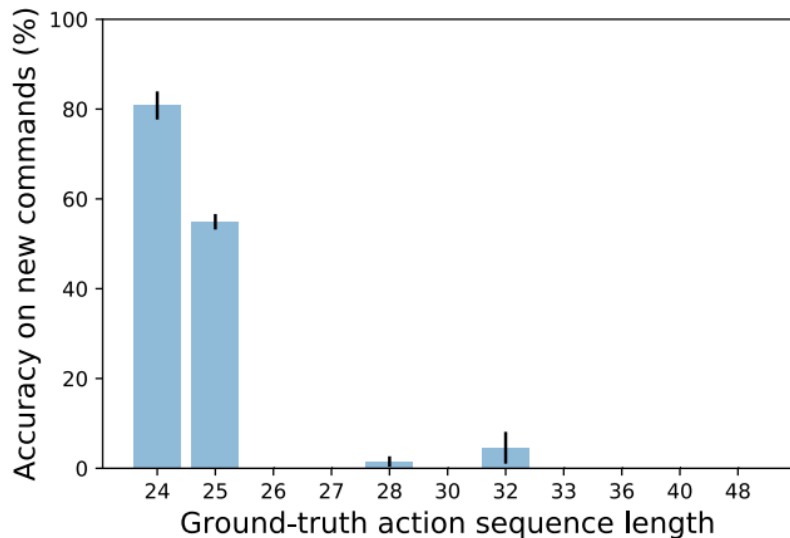
(Lake & Baroni, 2018; Ontanon et al., 2022)

jump	JUMP
run after run left	LTURN RUN RUN
look left twice and look opposite right	LTURN LOOK LTURN LOOK RTURN RTURN LOOK
<hr/>	<hr/>
jump twice after look	LOOK JUMP JUMP
turn left after jump twice	JUMP JUMP LTURN
jump right twice after jump left twice	LTURN JUMP LTURN JUMP RTURN JUMP RTURN JUMP

- On the Random Split, models get **~100%** accuracy
- However on the Add-jump split:
the best accuracy with **recurrent models** is **1.2%**
the best accuracy with **vanilla transformers** is **0.3%**

SCAN Experimental Results (subset): Length split

- Trained with target seqs of length <23
- Tested with target seqs of length >24
- Best accuracy with recurrent models is **20.8%**
- **Sharp drop at 25 for recurrent models!**
- Best accuracy with vanilla transformers is **<0.1%**



SCAN Experimental Results (subset): MCD split

(Keysers et al., 2020)

Dataset Split Method	SCAN	
	Random	MCD
LSTM+attention	99.9 \pm 2.7	6.1 \pm 2.2
Transformer	100.0 \pm 0.0	1.1 \pm 0.5
Universal Transformer	99.9 \pm 0.2	1.2 \pm 0.7

COGS: Compositional Generalization Challenge based on Semantic Interpretation

(Kim & Linzen, 2020)

Task: Natural language translation to logical form

Forms of generalization:

- Novel combinations of known primitives & grammatical roles
- Novel combination of modified phrases & grammatical forms
- Deeper recursion
- Verb argument structure alternation
- Verb class

COGS Examples

TRAINING

[[The girl]] = $\iota x. girl'(x)$, [[The cat]] = $\iota x. cat'(x)$, [[The boy]] = $\iota x. boy'(x)$

[[The cat loves the girl]] = $love'(\iota x. cat(x), \iota x. girl'(x))$

[[The hedgehog sees the cat]] = $see'(\iota x. hedgehog'(x), \iota x. cat'(x))$

GENERALIZATION

[[The boy loves the hedgehog]] = $love'(\iota x. boy'(x), \iota x. hedgehog(x))$

Case	Training	Generalization
S.3.3. Deeper Recursion		
Depth generalization: Sentential complements	Emma said that Noah knew that the cat danced.	Emma said that Noah knew that Lucas saw that the cat danced.
Depth generalization: PP modifiers	Ava saw the ball in the bottle on the table .	Ava saw the ball in the bottle on the table on the floor .

COGS Experimental Results (subset)

Model	Dev.	Test	Gen.
Transformer	0.96	0.96	0.35 (± 0.06)
LSTM (Bi)	0.99	0.99	0.16 (± 0.08)
LSTM (Uni)	0.99	0.99	0.32 (± 0.06)

CFQ: Compositional Freebase Questions

(Keyzers et al., 2020)

Task: Natural language questions to SparQL queries

Forms of generalization:

- MCD splits – similar atom distribution, but different compound distribution,

```
"question": "Did Agustin Almodovar executive produce Deadfall"
```

```
"sparql": "SELECT count(*) WHERE {\nns:m.041hs01 ns:film.producer.films_executive_produced ns:\nm.0gx0plf\n}"
```


CFQ Experimental Results (subset)

Dataset	CFQ	
	Random	MCD
LSTM+attention	97.4 \pm 0.3	14.9 \pm 1.1
Transformer	98.5 \pm 0.2	17.9 \pm 0.9
Universal Transformer	98.0 \pm 0.3	18.9 \pm 1.4

PCFG: Probabilistic Context-Free Grammar

(Hupkes et al., 2020)

Task: Commands on letters translated to resulting letter sequences

Forms of generalization:

- Systematicity, Productivity, Substitutivity, Localism, Overgeneralization

repeat A B C	→	A B C A B C
echo remove_first D K , E F	→	E F F
append swap F G H , repeat I J	→	H G F I J I J

PCFG Experimental Results (subset)

Experiment	LSTMS2S	ConvS2S	Transformer
Task accuracy*	0.79 ± 0.01	0.85 ± 0.01	0.92 ± 0.01
Systematicity*	0.53 ± 0.03	0.56 ± 0.01	0.72 ± 0.00
Productivity*	0.30 ± 0.01	0.31 ± 0.02	0.50 ± 0.02
Substitutivity, <i>equally distributed</i> †	0.80 ± 0.00	0.95 ± 0.00	0.98 ± 0.00
Substitutivity, <i>primitive</i> †	0.60 ± 0.01	0.58 ± 0.01	0.90 ± 0.00
Localism†	0.46 ± 0.00	0.59 ± 0.01	0.54 ± 0.02
Overgeneralisation*	0.68 ± 0.04	0.79 ± 0.06	0.88 ± 0.07

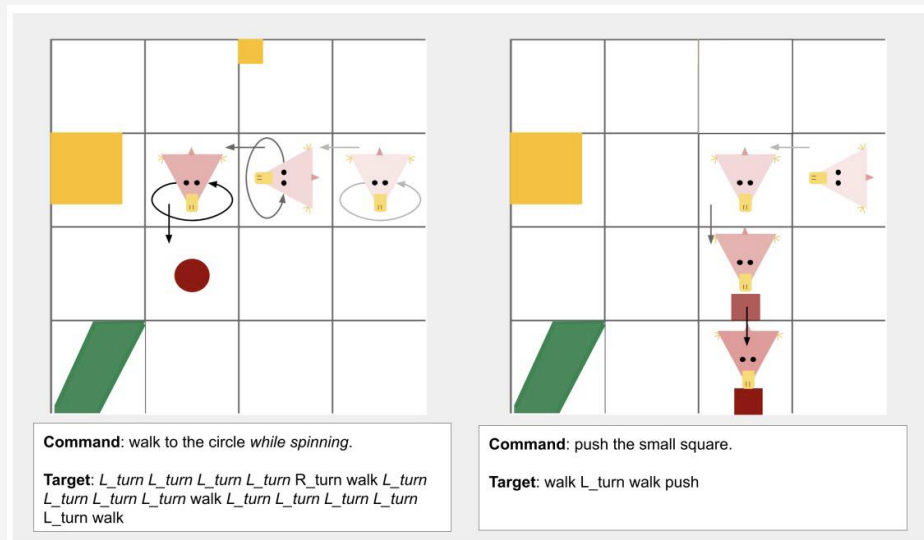
gSCAN: Grounded SCAN

(Ruis et al., 2020)

SCAN is not a grounded dataset, i.e., there is no context.

This paper provides a synthetic dataset where the meaning is grounded to a grid world with an agent.

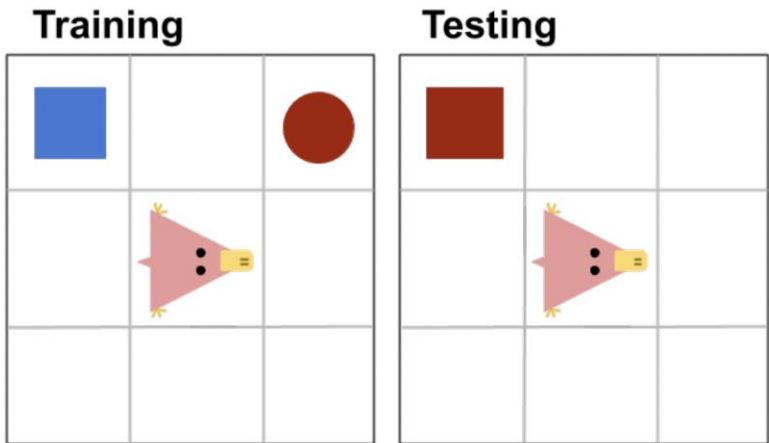
Output action sequence depends not only on the instruction but heavily on the context.



gSCAN: Grounded SCAN

Novel Composition of Object Properties : the (color, shape) combination has not been seen before

A sequence-to-sequence BiLSTM, with a visual encoder gets only ~24% accuracy on this split.



“Walk to the **blue square**.”

“Walk to the **red square**.”

“Walk to the **red circle**.”

“Push the **red square**.”

Course Plan

- What is Compositional Generalization?
- Compositional Generalization Benchmarks
- Approaches to Compositional Generalization



Approaches to Compositional Generalization

- Data augmentation
- Architectural modifications
- Pre-trained LLMs

Good Enough Compositional Generalization

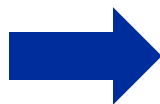
(Andreas, 2020)



Two discontinuous sentence fragments (underlined) which appear in *similar environments* (highlighted) are *substitutable*.
This can be applied to pairs of sentences for sequence-to-sequence tasks

Good Enough Compositional Generalization (GECA)

Training instances



Use GECA to create more instances



Train on original data + augmented data

The overall approach is linguistically crude and creates many sentences which are ungrammatical or not executable.

The hope is that creating a large number of such combined examples will help in generalization.

GECA generates 5% of the instructions for add primitive (**jump**) and 1% of the instructions for add template (**right**) automatically.

	jump / SCAN	right / SCAN
seq2seq	0.00 ± 0.00	0.00 ± 0.00
+ GECA	0.87 ± 0.02	0.82 ± 0.04

Demonstrates benefits on other tasks: Semantic Parsing on GeoQuery , Language Modeling and a few splits of grounded SCAN (gSCAN).

Related Approaches using Data Augmentation

Data Recombination for Neural Semantic Parsing (Jia & Liang, 2016):
Abstraction over entities and phrases to create more examples

Examples

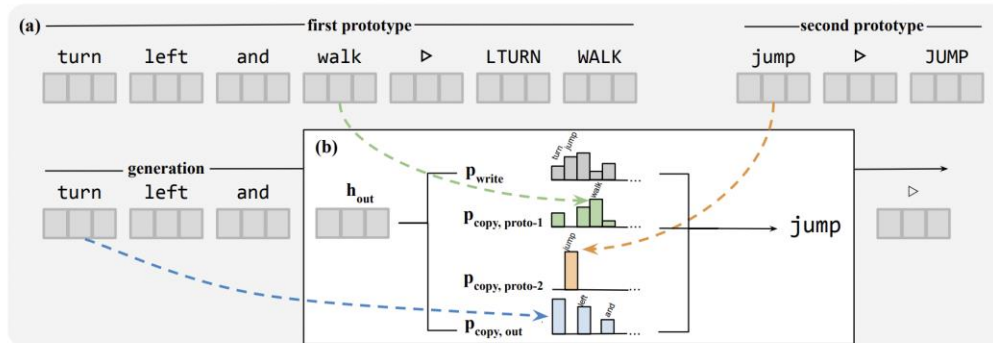
(“what states border **texas**?”,

answer(NV, (state(V0), next_to(V0, NV), const(V0, stateid(texas))))))

(“what is the highest mountain in **ohio**?”,

answer(NV, highest(V0, (mountain(V0), loc(V0, NV), const(V0, stateid(ohio))))))

Learning to Recombine and Resample Data for Compositional Generalization (Akyürek et al., 2021)



Architectural Modification: Disentangled Representations

(Li et al., 2019; Russin et al., 2019, 2020)

Systematic generalization in humans have been linked to the mechanism to separately process syntax from meaning of individual words.

(Chomsky, 1957; Fodor and Pylyshyn, 1988)

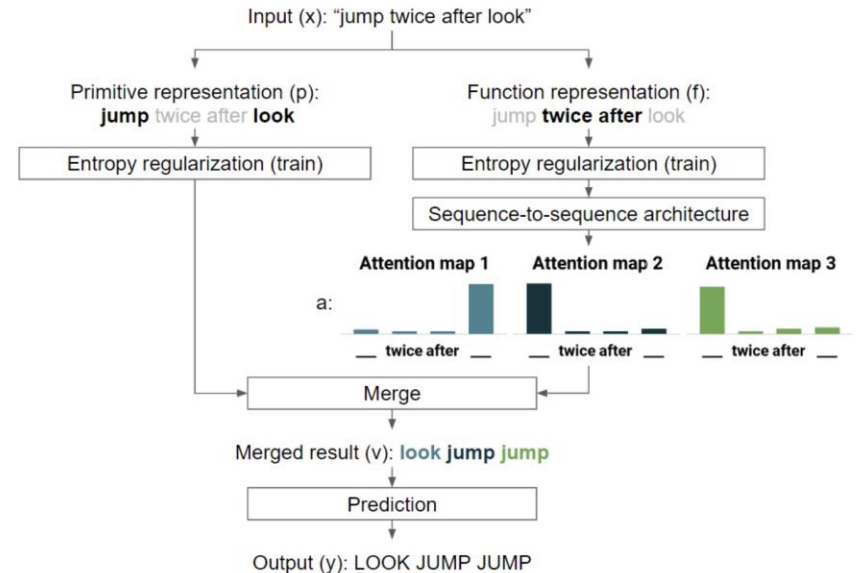
Question: *Can introducing this inductive bias help neural networks ?*

Two representations:

- Syntactic structure, related to the alignment of the words in the input and the actions in the output, captured by one representation.
- Translation of individual input words into actions, captured by the other.

Disentangling Syntax from Semantics

- Learn two representations for the input: one generates attention maps and the other maps attended input words to output symbols.
- Reduce entropy in each representation
- Output action type depends on one representation and output action order depends on another.



Disentangling Syntax from Semantics: SCAN Experiments

Do disentangled representations help models generalize to unseen primitives (actions) ?

SCAN primitive tasks ("jump", "turn left")

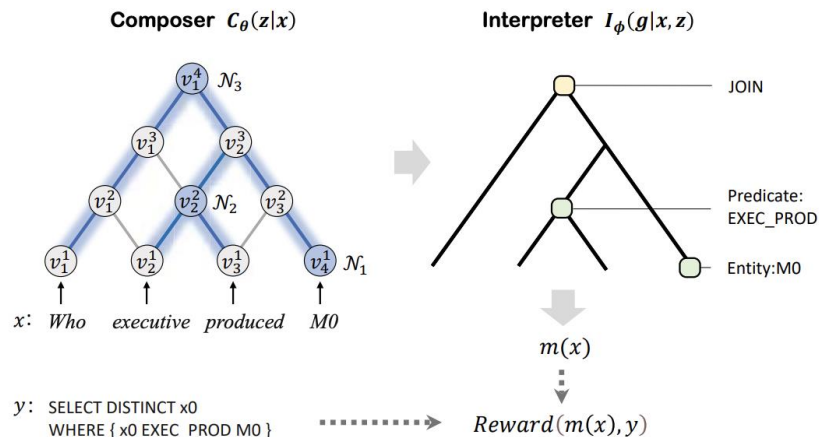
- "jump" task: their method dramatically improves accuracy to 99% from the 1% accuracy of the best model from Lake and Baroni, 2018
- "turn left" task: their method gets around 100% accuracy (best model from Lake and Baroni, 2018 gets 90%)
- Length split: their performance is unchanged relative to the best model from Lake and Baroni, 2018

Other Architectural Modifications

- Li et al, 2019 focuses on *lexical recombination*.
- Other works on SCAN, extends this to *algebraic recombination*, allowing for length generalization.
- Compositional generalization via neural-symbolic stack machines (Chen et al, 2020).
- Compositional generalization by learning analytical expressions (Liu et al, 2020).

LEAR: Learning Algebraic Recombination for Compositional Generalization (Liu et al., 2021) proposes a more general method, demonstrating its benefits on COGS and CFQ.

Model	Acc
Transformer (Kim and Linzen, 2020)	35 ± 6
LSTM (Bi) (Kim and Linzen, 2020)	16 ± 8
LSTM (Uni) (Kim and Linzen, 2020)	32 ± 6
LEAR	97.7 ± 0.7



The composer is a neural network (TreeLSTM) that produces the latent syntax tree z of input expression x . The interpreter assigns a semantic operation for each non-terminal node in z .

CPG gets perfect accuracy on COGS (to be presented by Tim Klinger tomorrow)

Pre-training in Compositional Generalization

(Furrer et al, 2021)

Contrasts a variety of specialized architectures with pre-trained Seq2Seq models (T5 family) on SCAN and CFQ splits.

Pre-trained LLMs help in compositional generalization but does not solve it.

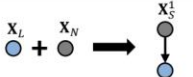
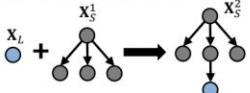
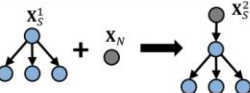
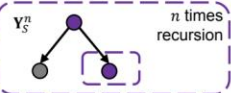
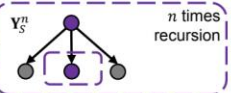
Pre-training **primarily helps in substituting similar words or phrases** (such as the add-jump split), while it **helps less on more complex phrases**. In particular it **hurts length generalization**, which requires further investigation.

Model	Add jump	Add turn left	Jump around right	Around right	Opposite right	Right	Length	SCAN MCD
LSTM	0.1	90.3	98.4 \pm 0.5	2.5 \pm 2.7	47.6 \pm 17.7	23.5 \pm 8.1	13.8	-
LSTM+A	0.0 \pm 0.0	82.6 \pm 8.2	100.0 \pm 0.0	0.0 \pm 0.0	16.5 \pm 6.4	30.0 \pm 7.8	14.1	6.1 \pm 1.7
CNN	69.2 \pm 9.2	-	-	56.7 \pm 10.2	-	-	0.0	-
GRU	12.5 \pm 6.6	59.1 \pm 16.8	-	-	-	-	18.1	-
GRU-dep	0.7 \pm 0.4	90.8 \pm 3.6	-	-	-	-	17.8	-
Transformer	1.0 \pm 0.6	99.6 \pm 0.8	100.0 \pm 0.0	53.3 \pm 10.9	3.0 \pm 6.8	92.0 \pm 15.1	0.0	0.9 \pm 0.3
Univ. Trans.	0.3 \pm 0.3	99.4 \pm 1.4	100.0 \pm 0.0	47.0 \pm 10.0	15.2 \pm 13.0	83.2 \pm 18.2	0.0	1.1 \pm 0.6
Evol. Trans.	0.6 \pm 0.6	100.0 \pm 0.0	100.0 \pm 0.0	30.2 \pm 28.4	11.6 \pm 14.6	99.9 \pm 0.3	19.8 \pm 0.0	1.6 \pm 0.6
Syn-att	91.0 \pm 27.4	99.9 \pm 0.2	98.9 \pm 2.3	28.9 \pm 34.8	10.5 \pm 8.8	99.1 \pm 1.8	15.2 \pm 0.7	-
CGPS	98.8 \pm 1.4	99.7 \pm 0.4	100.0 \pm 0.0	83.2 \pm 13.2	89.3 \pm 5.5	99.7 \pm 0.5	20.3 \pm 1.1	2.0 \pm 0.7
Equivariant*	99.1 \pm 0.0	-	-	92.0 \pm 0.2	-	-	15.9 \pm 3.2	-
GECA*	87.0 \pm 1.0	-	-	82.0 \pm 4.0	-	-	-	-
LANE	100.0	-	-	100.0	-	-	100.0	100.0
Meta seq2seq*	99.9	-	-	99.9	-	-	16.6	-
Synth*	100.0	-	-	100.0	-	-	100.0	-
NSEN	0.0 \pm 0.0	0.0 \pm 0.0	0.0 \pm 0.0	0.0 \pm 0.0	0.0 \pm 0.0	0.0 \pm 0.0	0.0 \pm 0.0	1.7 \pm 0.9
T5-small-NP	1.4 \pm 0.8	45.7 \pm 15.4	100.0 \pm 0.0	5.3 \pm 4.6	30.5 \pm 8.7	44.6 \pm 11.2	19.4 \pm 0.8	0.9 \pm 0.5
T5-small	84.1 \pm 1.0	73.0 \pm 5.8	100.0 \pm 0.0	31.8 \pm 1.0	58.2 \pm 10.4	88.7 \pm 8.9	10.9	6.9 \pm 1.1
T5-base	99.5 \pm 0.0	62.0 \pm 0.9	99.3 \pm 0.3	33.2 \pm 0.5	99.2 \pm 0.2	73.5 \pm 1.8	14.4	15.4 \pm 1.1
T5-large	98.3	69.2	99.9	46.8	100.0	91.0	5.2	10.1 \pm 1.6
T5-3B	99.0	65.1	100.0	27.4	90.0	76.6	3.3	11.6
T5-11B	98.3	87.9	100.0	49.2	99.1	91.1	2.0	9.1

In-Context Examples for Compositional Generalization

(An et al., 2023)

For each test example, they select the corresponding in-context examples, such that the primitives are covered and it is structurally similar to the test example.

Category	In-Context Examples	Test Case	Illustration of Combination
Primitive Substitution	input: shark output: SHARK input: A girl drew the boy . output: DRAW (GIRL , BOY , NONE)	input: The shark drew a boy . output: DRAW (SHARK , BOY , NONE)	
Primitive Structural Alternation	input: The goose baked . output: BAKE (GOOSE , NONE , NONE) input: A teacher noticed a chicken . output: NOTICE (TEACHER , CHICKEN , NONE)	input: A teacher baked the chicken . output: BAKE (TEACHER , CHICKEN , NONE)	
Phrase Recombination	input: Logan mailed Stella the cake in the pile . output: MAIL (LOGAN , IN (CAKE , PILE) , STELLA) input: The goose rolled a baby in a room . output: ROLL (GOOSE , IN (BABY , ROOM) , NONE)	input: A visitor in the pile rolled a resident . output: ROLL (IN (VISITOR , PILE) , RESIDENT , NONE)	
Longer Chain	input: The boy admired that Noah confessed that \ Emma was given a cookie . output: ADMIRE (BOY , NONE , NONE) \ CCOMP CONFESS (NOAH , NONE , NONE) \ CCOMP GIVE (NONE , COOKIE , EMMA)	input: The girl wished that a crocodile declared that \ the boy admired that Emma liked that \ Evelyn was passed a drink . output: WISH (GIRL , NONE , NONE) \ CCOMP DECLARE (CROCODILE , NONE , NONE) \ CCOMP ADMIRE (BOY , NONE , NONE) \ CCOMP LIKE (EMMA , NONE , NONE) \ CCOMP PASS (NONE , DRINK , EVELYN)	
Deeper Nesting	input: Noah appreciated a girl in a house \ beside the chair . output: APPRECIATE (NOAH , \ IN (GIRL , \ BESIDE (HOUSE , CHAIR \)) , NONE)	input: A dog painted the girl beside the chair \ in a house beside a road on a dish . output: PAINT (DOG , \ BESIDE (GIRL , \ IN (CHAIR , \ BESIDE (HOUSE , \ ON (ROAD , DISH \)))) , NONE)	

In-Context Examples for Compositional Generalization

Model	Setting	PrimSubs	PrimAlte	PhraReco	LongChain	DeepNest	Avg. Acc
code-davinci-002	Primitive Coverage	92.2	77.1	60.8	62.1	12.3	60.9
	+ Structural Similarity	99.8	99.7	65.3	87.0	26.0	75.6
text-chat-davinci-002	Primitive Coverage	92.2	75.4	47.0	65.0	6.3	57.2
	+ Structural Similarity	99.5	99.3	53.4	87.7	18.9	71.8
text-davinci-002	Primitive Coverage	88.5	66.4	38.7	46.5	2.9	48.6
	+ Structural Similarity	99.7	99.4	39.4	80.2	12.7	66.3
code-cushman-002	Primitive Coverage	82.6	55.6	21.3	29.3	5.0	38.8
	+ Structural Similarity	98.9	99.0	28.5	64.0	15.1	61.1
code-cushman-001	Primitive Coverage	76.6	60.7	16.9	5.0	1.0	32.0
	+ Structural Similarity	99.1	98.4	20.7	11.1	8.9	47.6
davinci	Primitive Coverage	69.4	52.3	9.4	2.3	0.2	26.7
	+ Structural Similarity	97.5	95.4	12.3	13.4	1.4	44.0
Fine-Tuning Baseline	-	93.6	97.9	14.0	5.4	0.0	42.2

Their results show that *smart* prompt exemplar selection can out-perform a fine-tuned baseline (GPT2-Large). However there is still a significant gap compared to the Neuro-symbolic approach on COGS (LEAR) which achieves ~98% accuracy.

Summary

- Compositional generalization is **of high importance in linguistics**
- Various recent **benchmarks help measure** compositional generalization of models
- **Off-the-shelf models** show **weak performance** on these benchmarks
- **Data augmentation** techniques can **improve** these models, but **only for special forms** of compositional generalization
- **Pretrained models** also show **improvements**, but also **for special forms**
- **Neuro-symbolic methods** with special **architectural inductive biases** are able to **better compositionally generalize**

Thank you! Questions?

Soham Dan
Parikshit Ram
IBM Research

Soham.Dan@ibm.com
Parikshit.Ram@ibm.com

- Partee, Barbara. "Lexical semantics and compositionality." *An invitation to cognitive science: Language* 1 (1995): 311-360.
- Pagin, Peter, and Dag Westerståhl. "Compositionality I: Definitions and variants." *Philosophy Compass* 5.3 (2010): 250-264.
- Hupkes, Dieuwke, et al. "Compositionality decomposed: How do neural networks generalise?." *Journal of Artificial Intelligence Research* 67 (2020): 757-795.
- Lake, Brenden, and Marco Baroni. "Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks." *International conference on machine learning*. PMLR, 2018.
- Kim, Najoung, and Tal Linzen. "COGS: A compositional generalization challenge based on semantic interpretation." *Conference on Empirical Methods in Natural Language Processing, EMNLP 2020*. Association for Computational Linguistics (ACL), 2020.
- Keysers, Daniel, et al. "Measuring Compositional Generalization: A Comprehensive Method on Realistic Data." *International Conference on Learning Representations*. 2020.
- Andreas, Jacob. "Good-Enough Compositional Data Augmentation." *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 2020.
- Jia, Robin, and Percy Liang. "Data Recombination for Neural Semantic Parsing." *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2016.
- Akyürek, Ekin, Afra FeYZa Akyürek, and Jacob Andreas. "Learning to Recombine and Resample Data For Compositional Generalization." *International Conference on Learning Representations*. 2020.
- Chomsky, Noam. *Syntactic structures*. Mouton de Gruyter, 2002.
- Fodor, Jerry A., and Zenon W. Pylyshyn. "Connectionism and cognitive architecture: A critical analysis." *Cognition* 28.1-2 (1988): 3-71.
- Furrer, Daniel, et al. "Compositional generalization in semantic parsing: Pre-training vs. specialized architectures." *arXiv preprint arXiv:2007.08970* (2020).
- An, Shengnan, et al. "How Do In-Context Examples Affect Compositional Generalization?." *arXiv preprint arXiv:2305.04835* (2023).

