Grounded SCAN Human: A Benchmark for Zero-Shot Generalizations

by

Dylan Sleeper

B.S. Computer Science and Engineering
Massachusetts Institute of Technology (2021)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2022

© Massachusetts Institute of Technology 2022. All rights reserved.

Author .....
Department of Electrical Engineering and Computer Science
January 14, 2022

Certified by .....
Boris Katz
Principal Research Scientist
Thesis Supervisor

Accepted by .....
Katrina LaCurts
Chair, Master of Engineering Thesis Committee

Grounded SCAN Human: A Benchmark for Zero-Shot Generalizations

by

Dylan Sleeper

Submitted to the Department of Electrical Engineering and Computer Science
on January 14, 2022, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

In this work, we collect a new human annotated dataset called Grounded SCAN Human (gSCAN Human) as an extension of the original Grounded SCAN (gSCAN) dataset. The original gSCAN dataset was created to test various compositional generalizations by holding out certain examples during train time. During test time, models must zero-shot execute commands that require the agent to move in new directions, commands that contain novel combinations of objects and adjectives, and other such generalizations in different test sets called splits. However, gSCAN does not contain splits that test zero-shot generalizations to new sentence structures and a whole new vocabulary. The gSCAN Human dataset was created to test these generalizations: can a model trained using a simple grammar generalize to human annotations? We collected and verified a total of $1,391$ human annotations across all of the gSCAN splits (excluding the test and dev split) and evaluated various models on each of the splits. We test the original gSCAN baseline with several modifications, including the baseline with a transformer replacing the encoder, and one with early multimodal fusion of the sentence encoding with the visual embedding. We also test a multimodal transformer similar to VilBERT, which is the state of the art on the original gSCAN splits. We find that the models are somewhat robust to varying sentence structure and new vocabulary; however the models are far less successful given a combination of the two, as evaluated by the human data.

Thesis Supervisor: Boris Katz
Title: Principal Research Scientist

Acknowledgments

First, I would like to thank my thesis supervisor Boris Katz for helping me through this entire process. I would also like to thank Andrei Barbu for helping me come up with my thesis topic and for providing useful insights on various aspects of my work. Next I must thank Candace Ross for implementing the multimodal transformers and helping me train, test, and debug them. I'd also like to thank Christopher Wang for helping me setup the gSCAN Human HITs, helping me verify them, and providing overall guidance throughout my Meng.

Lastly, I am so grateful for all of my friends and family. Thank you for giving me constant support and inspiration throughout not just my masters, but also my undergrad. This thesis would not have been possible without all of your support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Accurate and consistent generalization to out-of-distribution data is one of the toughest problems in natural language processing [16]. Models often fail to make simple generalizations that humans would find trivial [4][10][20]. If I am told to pick up the cup with a dragon on it, I can easily do so even if I have never seen such a cup. Regardless of the noun, adjective, and adverb combination, people can easily form novel combinations of the three and understand what they mean. This process is known as zero-shot generalization. Being able to perform such generalizations is essential when deploying machine learning models in the real world, especially if these models are intended to be deployed on robots that interact with people in everyday settings. Ultimately, having robots that communicate with humans through language is the goal; however, without models being able to consistently generalize, such a dream cannot become a reality. Even with the recent success of large language models like BERT [3], this problem is still not trivially solved [6]. Acquiring a training set that comprehensively covers the space of all possible commands that could be given to a robot is nearly impossible. Ideally, the training set could be composed of some subset of this space, and the model could generalize to new commands outside of that space.

The Grounded SCAN (gSCAN) dataset was created to address some of these issues in natural language models [20]. The dataset consists of a set of machine generated commands that are generated by fully enumerating a CFG (Context Free Grammar).

For each command, an example of the correct execution is generated in a simple grid world consisting of an agent and sets of objects that the agent must interact with. The model is then shown a specifically selected subset of the possible commands during train time, and is forced to generalize to the held out commands during test time. Different generalization conditions are tested through different divisions of the dataset called splits. The main shortcoming of this dataset is that all its sentences are generated from a CFG, so they lack variety in the way the sentences are structured and the vocabulary used. However, people will not be constrained to such a simple grammar when communicating with robots. To resolve this issue, we introduce a new dataset, gSCAN Human. gScan Human is a dataset composed of a randomly sampled subset of the gSCAN splits, with human annotations for each example. The dataset was collected by putting up assignments for annotators on Amazon Mechanical Turk. We showed people an execution trace and ask them to write a command that describes the actions of the agent. After all of the results were collected, we manually verified that each of the commands written were grammatically correct and followed the desired constraints of each individual split.

Through this new human dataset, we aim to evaluate different models and two types of embeddings: standard word embeddings (GloVe [17]) and contextual word embeddings (BERT [3]). These embeddings are both generated through purely textual training data. Given this, we aim to test whether or not these embeddings are properly grounded, i.e. whether they accurately encode the physical features of different nouns, adverbs, and adjectives. This is not quite as simple as ensuring that two words are "similar" or are sufficiently close in the embedding space. It tests whether or not these embeddings actually encode the proper features of given words. Take the two words "square" and "box" for example; one is a shape and the other is a physical object, so they likely do not lie very close to each other in the embedding space. However, if we are referring to the shape of an object, "box" and "square" are very similar as they both refer to a rectangular shape. If a model properly understood this, it could deduce that an object being referred to as "the box", is likely the same

object it has previously seen referred to as "the square." By testing whether or not a model is able to make such generalizations, we aim to analyze whether or not the embeddings encode this information in a way that can be captured by various models.

We model the problem as a Seq2Seq problem, and test with multiple variations of the encoder decoder architecture. The first models are variations on the baseline proposed in the original gSCAN paper [20]. The baseline consists of an LSTM encoder and decoder, with a CNN to encode the initial world state. We replace the learned word embeddings with GloVe embeddings for the first model, and for the second, the LSTM encoder is entirely replaced with a pretrained transformer known as BERT [3]. Using BERT enables the decoder to leverage BERT's pretraining to improve generalization to the human data. The second set of models are transformer only models with cross modal attention similar to VilBERT and other models of its kind [13][22]; these types of transformers are known as multimodal transformers. We use cross modal attention instead of unimodal attention, because it has been shown to outperform transformers with unimodal attention on most of the gSCAN splits. [18].

During training time, the models only see the original gSCAN training split. During test time, they then must be able to generalize to not only the original gSCAN splits, but also the human data. In addition to those splits, we create a separate set of splits to individually test certain variations introduced by the human dataset, in an attempt to understand exactly what features of the human data affect the accuracy of models. The first set of new splits is a set of synonym splits, where certain words are replaced with words of the same meaning. The second set of splits tests each of the models' robustness to variations in the sentence structure. By testing each of these variations individually, we are able to further analyze where the models fail.

On the original gSCAN splits, the models perform about as expected, getting high scores on most of the solved splits while failing to generalize in the others. On the human dataset, the accuracy on the test split (a randomly sampled hold out of the

train set, so other than the vocabulary and sentence structure, this set is sampled from the training distribution) of the baseline with GloVe embeddings is 13.95%, the accuracy of the baseline with BERT is 19.18% and the accuracy of the multimodal transformer model is 23.84%. The transformer improves upon the LSTM; however, overall they both still have relatively low accuracy on all of the human splits. This appears to be for multiple reasons, as all of the models struggle with new simpler splits introduced as well. The goal of this work however, was not to create successful models, but to see where current approaches stand in their ability to make such dramatic generalizations.

Chapter 2

Related Works

Systematic generalization has been tested in the past using the SCAN dataset [11]. The big difference between this dataset and gSCAN, is that SCAN is not grounded in any environment [20]; the model only receives text commands as an input and is tasked with producing an action sequence purely based on that textual input. gSCAN on the other hand, grounds these action sequences by having an agent execute the command in some grid world with an additional visual component [9]. ReaSCAN is an even newer dataset aimed to test more complex and nuanced generalization conditions, but the complexities introduced make collecting human data for such a dataset inherently difficult. As such, for our purposes here, we stuck to the original gSCAN dataset.

Over the last year, multiple models have been developed that solve some of the gSCAN splits. Some are much more task specific than others [4][10] while others are more general [5][18][20]. Both Gao 2020 and Kuo 2020 cannot be tested on the gSCAN human dataset because they lack the ability to generalize to different synonyms due to the fact that their models rely on the constrained vocabulary of the original dataset. The most successful model at the time of writing, is a multimodal transformer encoder and decoder with cross modal attention [18]. We implement a similar multimodal transformer model with cross modal attention to test transformer based approaches [13][19]. While many of these approaches can make good systematic

generalizations, none of them have been tested on vocabulary entirely left out of the training distribution; gSCAN Human aims to test this.

For our approach, we considered multiple options. We first considered semantic parsing [2][12][24], which is a method that decouples the parsing and planning processes in an effort to mostly separate language learning from path planning. This has proved to be successful in certain NLP and planning tasks. In Marzoev 2020 [15], a form of semantic parsing is used to allow a model to generalize from commands generated using a grammar to human sentences. To transfer between the two, the human sentence is embedded using some Seq2Seq model and the resulting embedding is compared to all of the possible commands generated from the grammar. The command from the original grammar closest in the embedding space is then fed to the planner to generate a sequence of actions. While this is very similar to what we are doing, we decide to opt for an end-to-end approach so that our model can be applied to problems in which a grammar that enumerates all possible commands does not exist, but such generalizations still must be made.

Chapter 3

Grounded SCAN

The Grounded SCAN (gSCAN) dataset [20] is an extension of the original SCAN dataset [11], that tests a model's ability to make compositional generalizations in a grounded language setting.

3.1 The Environment

The environment is a grid world composed of an agent and different objects the agent can interact with. A visualization of the environment can be seen in Figure 3-1. The small pink triangle represents the agent, while all of the other shapes are different objects the agent can interact with. The agent is capable of six different actions: staying still, turning left, turning right, moving forwards, pushing, and pulling. The agent can pass through objects without moving them, but it also has the option to push them forwards or backwards by performing the push or pull action respectively.

Each object has three different attributes: shape, color, and size. There are three different shapes, four different colors, and three different sizes. Each object can be any combination of these three attributes. One important thing to note, is that the size affects how an agent can push or pull a given object. If an object is large, the agent must perform two push/pull actions on the object to move it one space. For smaller objects, the agent only has to perform these actions once to move the object

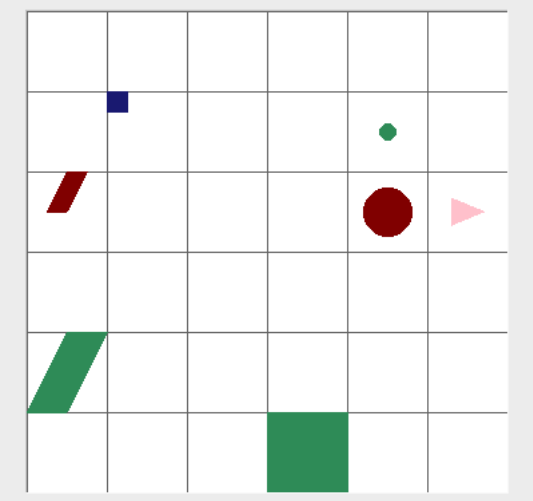


Figure 3-1: This is what the environment looks like. The pink triangle is the agent, and all of the other colored shapes in the scene are objects the agent can interact with.

a single space.

3.2 The Commands

Each command denotes how the agent should interact with a specific object in the environment. The commands always uniquely specify a single object and a single action that the agent must perform related to that object. The commands are generated from a CFG. By enumerating the entire CFG, we get every possible "valid" command. The commands are structured as follows. The first part of the command always specifies the action that should be done. This can be one of "Walk to the", "Push the", or "Pull the." The next part of the command is the target object. The target object is always specified by the shape, but in addition to that attribute, sometimes the color and size are also mentioned. Examples of target objects include "square", "blue circle", and "big red cylinder." The command either ends there, or also includes an adverb/adverb phrase that describes how the agent must perform the action. This includes "hesitantly", "cautiously", "while spinning", and "while zigzagging."

This is an efficient way to generate thousands of commands. However, it lacks variety

in multiple areas. The first area of concern is the structure of the sentences. By structuring the sentences in the same way every time, a model might overfit to that structure and may not be able to generalize to other sentences that do not follow the same pattern. For example, the model might be able to understand the command "Walk to the red square while spinning" but may fail to execute a command of the same meaning, such as "While spinning, walk to the red square." The second area of concern is the limited vocabulary. Lets take the previous command as an example. This command could also be specified as "Walk to the red square while **rotating**" or "**Go** to the red square while spinning." In this case, the model may overfit to the limited vocabulary it has been introduced to and be unable to generalize to new synonyms like "rotating" and "go." With gSCAN Human, we aim to evaluate whether or not such overfitting occurs when a model was trained on these constrained commands.

3.3 Splits

Once all of the commands are generated, the results are divided into separate splits to test certain generalization conditions. Commands containing certain objects, noun adverb combinations, and directions are held out in the train set. They are then introduced in the test sets, to see if the model can make generalizations based on what it has learned to correctly execute commands with some feature it has never seen. There are eight different splits which I will briefly explain.

Test Split (Test) The first split is identical to a traditional test set. A random sample of examples is held out from the training set and put into this test set. This split only evaluates how well the model learned the task without testing any unique generalization conditions.

Novel Composition (Yellow Squares, Red Squares) These two splits both test for novel combinations of objects and properties. The first split holds out all of the training examples where a yellow square of any size is mentioned. The model will

still see yellow squares as the target object, but they are only ever referred to by their shape and size but never by their color (i.e. as "the big square" or just "the square"). The second split holds out all examples that reference a red square. The difference between this and the aforementioned split, is that there are no examples in the training set that reference a red square as the target object. Due to this fact, this is more challenging than the first of the two splits.

Novel Direction (Novel Direction) The Novel Direction split does not test a language generalization, but a movement generalization. In this split, all examples that contain a target object that is located to the southwest of the agent are held out in training. At test time, the model must generalize to this new direction by combining its understanding of walking south and walking west. Every model thus far has been unable to make this generalization.

Novel Contextual References (Contextual) The Contextual split tests for the model's ability to physically understand sizes. Through this split, we want to see if the model truly understands what the size adjectives mean or if the model just forms a mapping between those adjectives and what objects it has seen with that attribute. In other words, we want to see if the model understands the relativity of sizes. To test this, all examples in which the smallest target object is size two are held out. There are scenes in which an object of size two exists, but it is never the target object when the command contains the adjective "small." At test time, the model is then shown examples in which the target object is the smallest object, that object is of size two, and command references that object as "small."

Novel Composition (Relativity) The Relativity split tests a different size generalization. In the gSCAN dataset, smaller objects are classified as "light" while larger objects are classified as "heavy." To move a heavy object, the push/pull action must be performed on the object twice for it to move a single space, unlike light objects, in which either of those actions only need to be performed once to move the

object. In the training set, all examples of a heavy square are held out but the model is still shown heavy instances of all other objects. At test time, the model must make the correct generalization to properly move heavy squares.

Novel Adverbs (Adverb 1, Adverb 2) These last two splits test how well the model can learn to apply adverbs to unseen combinations of adverbs and verbs. The first of the two splits tests this by giving the model a limited number of examples in the training set. The second of the two tests this by showing the model thousands of examples of the verb "pull" and "while spinning" without ever showing the model a command with both in them. These are two of the harder splits; most models fail to achieve high accuracy on either one.

Chapter 4

gSCAN Human

gSCAN Human is a subset of the original gSCAN dataset with examples that have been annotated by humans. By using human annotations, we address some of the missed generalization conditions in the original gSCAN dataset. We are now able to test another type of generalization: can a model trained on gSCAN's simple and formulaic sentences generalize to an out of distribution set of sentences of varying length, structure, and vocabulary?

4.1 Collecting the Data

To collect this dataset we used Amazon Mechanical Turk. This is a platform that enables "requesters" to put up assignments to be completed by "workers." To get an idea of what these assignments look like, see Figure 4-1. Due to some limitations of Mechanical Turk, we had to set up our own infrastructure to put up and monitor these assignments, also known as HITs.

For a worker to gain access to our HITs, they had to fill out a qualification survey to determine whether or not they understood the task. Without this survey, we often got incorrect responses that violated many of the instructions. The survey went through many iterations, and in the end, consisted of a detailed set of instructions and a set of questions at the end to verify that the worker had read the instructions

Figure 4-1: An example from one of the MTurk HITs. Depending on the split the example comes from, extra instructions are included as necessary. Each HIT contained 8 of these examples.

and properly understood the task. For a detailed look at the instructions, see Figure A-1 in the appendix. By completing the survey, workers were given a qualification that enabled them to access the task.

In the actual HIT, we showed workers a set of GIFs consisting of an action trace from one of the gSCAN splits and asked them to write the command that the agent was following. However, to conform with the various generalization conditions in the original splits, we often had to include some other instruction specific to a given example. For example, in some splits, the color of the target object is always mentioned in the commands. Whenever an example was part of such a split, we instructed the worker to mention the color of the target object. We also found that whenever the agent was moving in a unique way (i.e. zigzagging or spinning), workers would often neglect to mention this. So whenever an agent was moving in a unique way, implying that some adverb was present in the original command, we instructed the workers to mention how the agent was moving.

We wanted to collect data from as many unique workers as possible to capture a

wide variety of vocabulary and sentence structure. To do so, we limited the number of HITs a given worker could complete. This is unfortunately not a feature of Mechanical Turk, so we wrote a script that automatically went through all of the completed HITs and stored each of the responses along with some metadata associated with the HIT (i.e. worker id, assignment id, etc). The script then counted the number of HITs a given worker had completed, and if they had completed more than the desired number of HITs, their qualification was removed so that they could no longer access anymore of our HITs. We set this threshold to five HITs, but because there was some delay between a worker completing a HIT and this script recording that information, some workers were able to complete six.

After collecting this initial sample, we realized that the quality of the results incrementally decreased over time. We recognized that our task was rather challenging in many regards, so for a second round we selected workers that provided good responses in the first sample. We then had these workers complete as many HITs as they wanted. While this decreased the variety, it drastically increased the quality of our results.

To verify the data, we manually went through all responses and removed the ones that did not follow the instructions. Many researchers post secondary HITs to complete this task; however, we decided that the verification required too much specific knowledge about the original gSCAN dataset to properly verify the result. To deem whether or not a HIT was valid, we evaluated each response on the following criteria. First we made sure the sentence followed the extra instructions given. Second, we ensured that the command was specific enough to detail the exact task the agent was doing. For example, some workers wrote "Move to the blue square" in an example with multiple blue squares. Given the aforementioned command, the agent would have to guess which square was being specified. In cases where even we could not properly generate the correct sequence of actions 100% of the time given the worker's command, the command was thrown out.

	Min Length	Max Length	Avg Length	Unique Words
Original Dataset	3	7	5.21	18
Human Dataset	3	41	11.06	312

Table 4.1: Original vs Human Dataset

4.2 Results

In total we collected 1,391 samples from 65 unique workers. This corresponds to roughly 173 examples for each split with each worker annotating 21.4 examples on average. We did not collect examples from the train or dev splits because both splits would not be testing any generalizations already being evaluated in the test split. Note that some splits were more difficult than others, which is why the number of responses in each split varies. Workers tended to struggle when providing descriptions of the various types of movement. They either did not notice exactly how the agent was moving, or they simply failed to mention the alternative way in which the agent moved. Workers struggled the most with zigzagging and cautiously. For zigzagging, workers tended to ignore the movement, especially when the agent only moved up a row or two. For cautiously, workers often mistook the agent turning left then right to be spinning. Even though the instructions and the survey questions tried to ensure that such mistakes would not be made, it occurred rather frequently. For this reason, examples with those adverbs are less plentiful than others.

4.2.1 Comparisons to the Original Commands

The Human dataset is vastly different than the original gSCAN dataset. Specific statistics regarding both datasets can be seen in Table 4.1. First, the average sentence length of the commands in the Human dataset is much greater than the original gSCAN dataset. This is likely due to the fact that people generally over-specified what the agent was doing. For example, sometimes workers would say "Do a half spin at the beginning, then..." whenever the agent had to turn around to go towards an

object. Workers would also sometimes specify the objects the agent passed through on its way to the target object. While this information is unnecessary, it was not incorrect, so commands with extraneous information were kept. If workers were told exactly what to mention, such unnecessary observations could have been avoided. However, this would have greatly constrained the types of sentences in dataset. We likely would have limited the responses to those that were just slight variations on the original gSCAN dataset which we could have generated via a slightly more complex grammar. By keeping such sentences, we will test if our models can ignore or interpret these extra observations and still execute the command correctly.

As expected, there was also a larger variety of vocabulary used. Most of the variety in vocabulary came from the words used to describe the objects and the variations in movement. The object that was never mentioned using the vocabulary of the original gSCAN dataset was the "cylinder." This object was often called either a "parallelogram" or a "rhombus." I accepted either of these answers because they do mostly accurately describe the shape. Most of the other variety came from the descriptions of movement. Workers often described "hesitantly" as "slowly" or "pausing with every step", "zigzagging" was often described as "moving upwards left then right" or some variation on that, and "cautiously" was often described as "turning left then right on each step." While we tried to lead workers to use adverbs similar to those in the original gSCAN dataset through some of the questions in the qualification survey, most still used their own words to describe exactly how the agents were moving. Unlike both the objects and the adverbs, when describing actions, the vocabulary was relatively similar to the original gSCAN dataset. Generally people used walk, push, and pull to describe what the agent was supposed to do; however, there were some slight variations. Some examples of these variations include using "move" instead of "walk" and "drag" instead of "pull." We accepted sentences in which these variations still clearly specified what action should be completed.

4.3 New Datasets

To individually test different variations in the commands, we created multiple unique programatically generated splits based on the variations seen in the gSCAN Human dataset. By creating splits with only certain variations, we were able to test specifically what perturbations the models can and cannot generalize to. The first set of splits replaces individual words in the commands with synonyms; the length and structures of the sentences remain the same. This will test to see if the GloVe and transformer embeddings can transfer their pretrained knowledge to this task. The second set of splits varies the sentence structure to see if the models themselves are robust to such variations. More details regarding exactly what variations were tested can be found in Section 6.3.

Chapter 5

The Models

For this task, we propose multiple variations on the original baseline and a novel transformer architecture based on VilBERT and other joint vision language models. By leveraging cross-modal attention and the inherent knowledge of large language models, we hope to improve a model's ability to generalize. We also aim to account for the new generalization conditions being tested by gSCAN Human. Namely, the variety in sentence structure and the larger vocabulary.

5.1 The Baseline

The first model we use for evaluation is the baseline described in the original gSCAN paper. We use this architecture in order to compare the performance of traditional Seq2Seq models and transformers. This model is composed of a Bi-Directional LSTM [7][21] to encode the commands and a CNN to encode the grid world. Note that the entire grid world is fed into the model as a single "image" which differs from the transformer approach that I will discuss later. The "image" is also not represented as a set of pixels in this case, but a three dimensional matrix that acts as a one hot encoding for different object attributes. After both the image and text are embedded, both are fed into an LSTM decoder with attention to produce an action sequence [1]. The model only receives an image embedding of the first state, and must generate the entire action sequence conditioned on that first state, the command embedding,

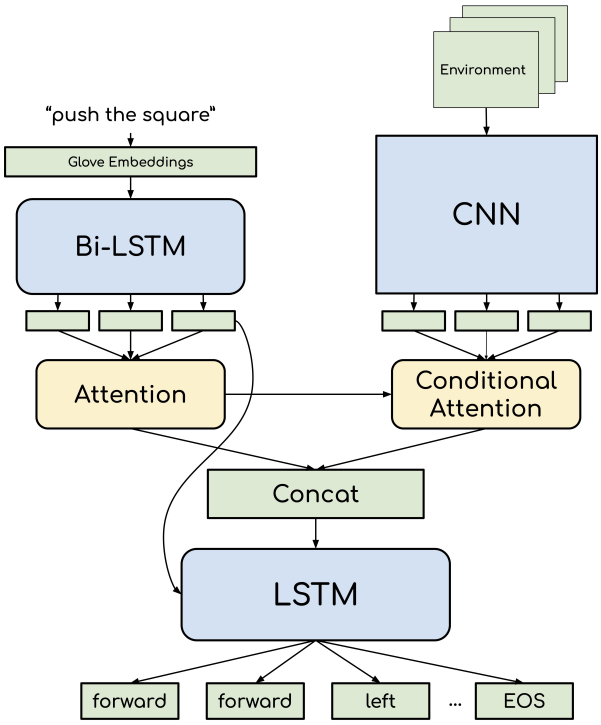


Figure 5-1: LSTM Baseline model

and all of its previous outputs. The model can be seen in Figure 5-1.

One change is made from the original baseline to be able to test the model on the new programmatically generated splits and the human dataset. The original baseline has a learned embedding layer with a small vocabulary consisting only of the words present in the original gSCAN dataset. Using the model in this state would make the required generalizations impossible, so the embedding layer is replaced with pre-trained GloVe embeddings [17]. The embeddings are then frozen during training time, such that the model can be evaluated on its ability to generalize to unseen words.

5.2 Transformer Models

Transformers are another model used for Seq2Seq tasks. They differ from all other traditional approaches by using attention as the main method of learning [23]. Unlike

other Seq2Seq models that rely on some form of an RNN with or without an attention mechanism, transformers learn by entirely relying on a type of attention called self attention. This self attention mechanism works as follows. Given some input sentence $(x_1, x_2, ..., x_n)$, transformers first embed each token through some trained embedding layer to produce continuous representations $(z_1, z_2, ..., z_n)$. A unique learned positional encoding is then added to each continuous representation. Without this, it would be impossible for the model to know the order of the words in the sentence because unlike an RNN, each word is passed into the model simultaneously.

Given the continuous representations of each token z_i, the model attends to each token using scaled dot product attention. For each token, a query q_i, key k_i, and value v_i is calculated. Each one is calculated by multiplying each input token by a learned matrix (one of W^Q, W^K, and W^V). Using these values, we calculate a set of $d_{i,j}$s then take softmax over j:

$$d_{i,j} = q_i \cdot k_j \tag{5.1}$$

$$s_i = Softmax(d_i) \tag{5.2}$$

To calculate the output of the attention a_i we then sum the results:

$$a_i = \sum_j v_j * s_{i,j} \tag{5.3}$$

An a_i is calculated for each token z_i and is then passed into the next layer. This process is then repeated for L layers. Additionally, most of these transformer models have multiple attention heads. For each layer, H values are calculated for each token using $3 * H$ different weight matrices (W_h^Q, W_h^K, and W_h^V), such that each attention head can attend to different parts of the sentence. The resulting tokens are all concatenated together, then scaled through a single linear layer to produce each a_i.

5.2.1 Transformer Encoder + Baseline Decoder

Our second model replaces the LSTM encoder in the baseline with a transformer. We initialize the transformer to the weights of a pretrained transformer, BERT [3]. BERT is a transformer that has been pretrained using BookCorpus [25] (800 million words) and Wikipedia (250 million words) on two unsupervised tasks: masked language modeling and next sentence prediction. By pretraining on these two tasks, the model has already seen hundreds if not thousands of instances of each word we will be testing with. So even if the model has not seen a given word in the gSCAN dataset, it should be able to perform some generalizations through transfer learning of its pretrained knowledge.

5.2.2 Transformer Encoder + LSTM Decoder

Our second transformer encoder + LSTM decoder model differs slightly from the previous one in that it tries to fuse the two modalities together before the embedding reaches the decoder [8][14]; this model can be seen in Figure 5-2. We fuse the two modalities before the decoder for a few reasons. First, it enables the model to ensure that just enough visual and textual information comes through into the final embedding. Second, it further decouples the encoding and decoding process. Decoupling the two helps ensure that most of the work done by the decoder is planning and not further encoding of the input command. This reasoning is similar to that behind semantic parsing for the purposes of planning, but in this case, the decoupling is a learned representation and is therefore less explicitly defined. The decoupling process is furthered by initializing the decoder hidden and cell values to a learned representation instead of using the encoded command. Using learned representations has been shown to improve generalization in some cases and can speed up training time. The last reason for fusing the two modalities before the decoder, is that such a process forces the embedding size to be much smaller. In the original baseline, the model is able to attend to all of the words in the command while in this case, the model is only given the [CLS] token from the BERT encoder (note that an extra layer is

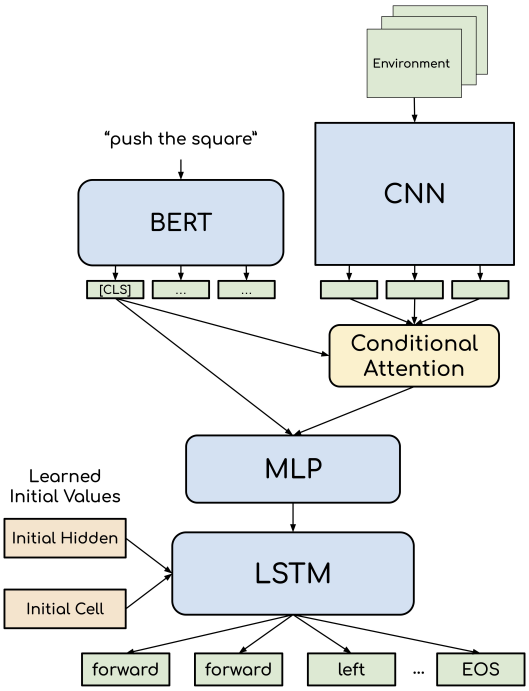


Figure 5-2: Transformer + LSTM Model with late fusion

added to BERT and fine tuned to create a richer sentence embedding). The hope is that by compressing the representation into a smaller size, the model will no longer overfit to specific features of the individual words. The previous model is much more expressive in that regard, but we hope that by limiting the expressiveness, the model will be able to generalize better to out-of-distribution test data.

5.2.3 Multimodal Transformer Encoder + Transformer Decoder

We also evaluate transformer models with cross modal attention. The architecture we use can be seen in Figure 5-3. Unlike in the baseline and the modified baseline, which both encode the command and image separately, cross modal attention enables the model to attend to the command and image simultaneously. This makes it such that the final embeddings of both the command and image are conditioned on both the vision and language inputs. This approach is inspired by VilBERT [13], a

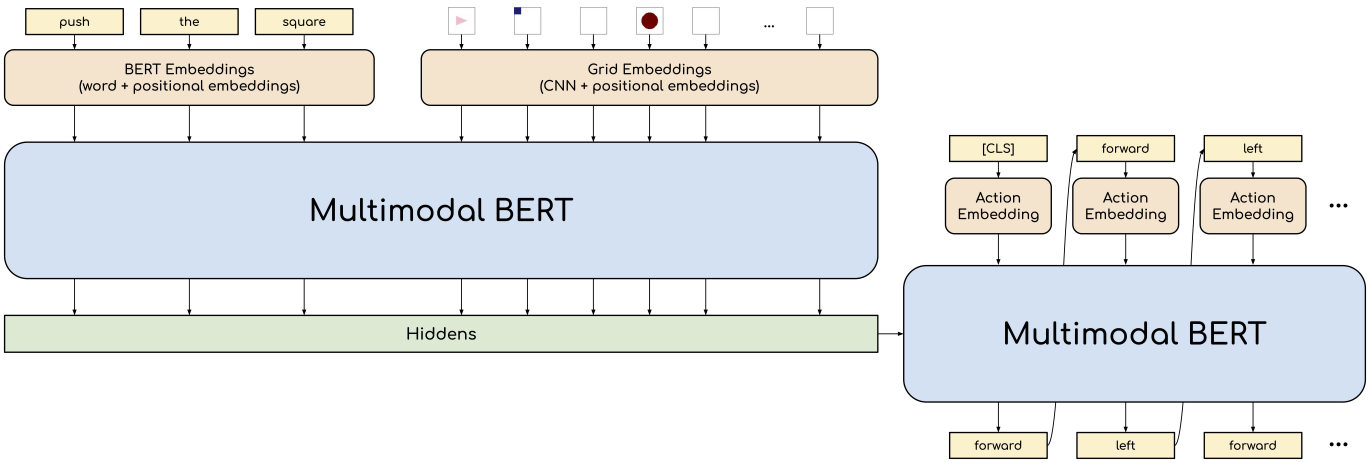


Figure 5-3: Transformer Encoder + Decoder

transformer-based model with cross modal attention.

Like the previous models, the model consists of an encoder and decoder. For the encoder, we start by dividing the first image into each individual grid cell which are then passed into the model as separate tokens. Each grid cell is passed through a CNN to produce an embedding g_i, and is then concatenated with a positional embedding to encode its position in the grid. The resulting image embeddings are then concatenated to the text embeddings z_i which enables our model to attend to both the command and the image together. The resulting attention outputs are passed to the decoder.

In the decoder, we feed each subsequent action into the model to autoregressively produce the entire action sequence A. The subsequent states are not fed into the decoder; it only receives the previous actions. In a deterministic environment like gSCAN, we do not need to give the model anything more than the start state and the actions it has taken so far. That alone is enough information to determine exactly where the agent and all of the objects are located at a given time step. This significantly reduces the length of the input sequence which speeds up training and lowers

the memory footprint when producing the final action sequence. This final action sequence is generated one action at a time. At a given time step t_c, the model can attend to any of the previous actions $a_{t<t_c}$ to produce an action a_{t_c}. After producing an action, the model is then also given that action; the process then repeats until the model produces an [EOS] (end of sequence) token.

Chapter 6

Experiments

6.1 Model Details

Baseline: The original model with GloVe embeddings of size 100. The encoder is a one layer bidirectional LSTM with a hidden size of 128. The decoder is a one layer LSTM with attention for both the text and images, also with a hidden size of 128.

B + T: The same architecture as the original baseline with the LSTM encoder replaced with a pretrained BERT with no fine tuning. In this case the hidden size of the decoder is increased to 256.

Modified B + T: A similar architecture to the previous B + T model with the differences as described in section 5.2.2. The decoder also has a hidden size of 256.

MMT Pretrained BERT: A fully multimodal transformer model as described in section 5.2.3. Both the encoder and decoder are the same architecture as the original BERT: a hidden size of 768, 12 attention heads, and 12 layers. The encoder is initialized with pretrained BERT weights while the decoder weights are randomly initialized.

MMT Pretrained Embeddings: Another fully multimodal transformer model that is much smaller than the previous. The hidden size is 256, there are only 4

	Baseline	B + T	Modified B + T	MMT Pretrained BERT	MMT Pretrained Embeddings
Test	97.30	93.45	97.41	96.74	**99.96**
Red Squares	16.31	61.05	33.29	**94.23**	93.34
Yellow Squares	5.13	47.80	18.54	97.56	**99.98**
Novel Direction	0.00	0.00	0.00	0.00	0.00
Relativity	39.26	33.60	39.26	92.90	**99.30**
Contextual	96.24	93.56	97.26	99.18	**99.89**
Adverb 2	17.24	19.11	18.95	**35.50**	22.18

Table 6.1: Exact match percentages on the original gSCAN splits

attention heads, and 4 layers. The weights for both the encoder and decoder are randomly initialized, with the exception of the initial embeddings used by the encoder are the pretrained BERT embeddings to enable some form of generalization.

6.2 Results on the original gSCAN Dataset

The results on the original gSCAN splits can be seen in Table 6.1. The first thing of note, is the difference in performance of the baseline with and without the transformer encoder on the red and yellow squares splits. Having a transformer instead of an LSTM as the encoder makes a huge difference on both of the two splits. Performance is four times better on the red squares split and is nine times better on the yellow squares split. Even without any fine tuning, the embeddings provided by the transformer are still far more effective than those of the LSTM encoder. The difference is not nearly as dramatic on the modified baseline with the transformer. It seems that the smaller sentence encoding and therefore the inability to attend to each word in the command, leads to lower performance on both of those splits. The single fine tuned BERT layer fails to encode "red square" and "yellow square" in a way that enables the decoder to make the proper generalizations. Having an attention layer conditioned on the hidden state likely enables the higher performing baseline with a transformer to make such generalizations better.

Both of the multimodal transformer models significantly outperform all of the variations on the baseline in nearly every split. Given the results presented in Qiu 2021

	Baseline	B + T	Modified B + T	MMT Pretrained BERT	MMT Pretrained Embeddings
Synonym Verb	70.56	91.95	**96.95**	45.47	64.40
Synonym Color	43.10	64.90	**79.78**	40.05	42.15
Synonym Shape	30.03	37.23	32.57	48.20	**52.87**
Synonym Size	34.55	65.30	**67.45**	48.40	36.30
Synonym Adverb	17.40	66.53	**69.73**	17.60	33.33
New Structure	60.93	48.60	48.40	**66.87**	52.27

Table 6.2: Exact match percentages on the new Synonym and Sentence Structure splits

[18] this is not surprising. They showed, through ablation studies, that cross modal attention makes a huge difference when it comes to generalization across most of the gSCAN splits. That is likely the reason behind the drastic difference between the baseline and multimodal transformer results in the red squares, yellow squares and relativity splits.

6.3 Results on the new splits

6.3.1 Synonym Splits

For each new synonym split, I took a single word and replaced it with one of two to five similar synonyms. I did this for all of the verbs (walk, push, pull), colors (red, green, blue, and yellow), shapes (square, circle, and cylinder), sizes (big and small), and adverbs (hesitantly, spinning, and zigzagging). The results can be seen in table Table 6.2.

The baseline is outperformed in every synonym split by both of the baselines with a transformer encoder: the biggest difference being in the color synonyms split. This is likely due to the nature of contextual embeddings provided by the transformer. Many of the color synonyms have multiple meanings (for example, one of the synonyms for green is lime) which can be captured by contextual embeddings but cannot be captured as well by GloVe embeddings. The transformer embeddings also greatly outperform the original baseline on the verb, size, and adverb splits. The reason

behind such a performance boost in these splits is less apparent. It is likely that the pretraining scheme in transformers simply creates better embeddings. Contextual embeddings also provide a nice boost even though these words do not have vastly different meanings. The resulting embeddings of the actual objects and verbs might be different due to the adjectives and adverbs respectively, something a transformer can more easily model than an LSTM due to its self attention mechanisms.

Strangely, even though the multimodal transformer model is able to outperform the baseline and baseline with a transformer on all of the original gSCAN splits, the same is not the case for all of the new splits. In fact, the entirely multimodal transformer models perform much worse compared to the modified baselines on most of the splits. We even see for the simple binary differentiation between push and pull, the model fails to make a clean generalization. In the case of the adverb splits, the multimodal transformer models always guess "zigzagging" for any of the synonyms. The models seem to have overfit, which could be due to the number of large number of parameters in both the encoder and decoder. Transformers need far more data than more traditional Seq2Seq models while also requiring more parameters to be as or more successful. While they scale in number of parameters far better than RNNs, and outperform RNNs on most NLP benchmarks, the amount of data they require is also far greater. Such a result goes back to the discussion in Section 5.2.2. Transformer models of this size are far more powerful and expressive to their single layer LSTM counterparts. For such generalizations, that extra expressiveness may be more of a detriment than benefit. The generalizations being made in the synonym splits are quite simple, especially in both the push and pull and the big and small instances. Having such a complex model learn a general solution to this problem is rather difficult, while such a solution is more easily learned by a smaller less expressive model. This is further demonstrated by the comparison between the results of both baselines with transformer encoders. The model with multimodal fusion, that compresses the entire sentence into a single vector before passing it to the decoder, outperforms all of the other models in four out of five of the synonym splits: the difference being most

evident in the color split. By compressing the sentence into a single vector, we force the model to learn simpler and more general relationships. While this is somewhat detrimental in some of the original gSCAN splits, it is beneficial to the models ability to perform zero-shot generalizations across a new vocabulary.

6.3.2 Sentence Structure Splits

The other set of new splits is the set that varies the sentence structure of the commands. A few different variations are applied to the sentence structure of the commands. In the first split, the location of the adverb is flipped. Instead of placing the adverb at the end of the sentence, it is moved to the beginning. For example, the sentence "Push the red square while spinning" is changed to "While spinning, push the red square." In the second split, the push and pull commands are broken up into two parts. The first clause mentions moving to the target object, and the second mentions the action that should be performed on the target object. In this split the sentence "Push the red square" is changed to "Move to the red square then push the red square." The final split is similar to the last one, except the final mention of the object is replaced with "it", so the aforementioned command turns into "Move to the red square then push it." The results from these splits can be seen in Table 6.2.

In the sentence structure splits, the transformer model initialized with a pretrained BERT greatly outperforms the baseline models with transformers as well as the smaller transformer model. This is quite peculiar, as it is contradictory to most of the other results we have seen so far. It seems as if more parameters make for worse generalizations. However, in the case of sentence structures, the largest model seems to generalize the best. This is likely due to the fact that it is initialized with a pretrained BERT, a model that has been trained with corpora containing millions of words. All of that pretraining with additional fine tuning likely enabled the model to generalize to differing sentence structures better than any of the other models.

	Baseline	B + T	Modified B + T	MMT Pretrained BERT	MMT Pretrained Embeddings
Test	13.95	16.86	19.18	15.70	**23.84**
Red Squares	3.00	13.17	10.18	14.97	**16.17**
Yellow Squares	4.10	15.20	5.26	23.39	**25.73**
Novel Direction	0.00	0.00	0.00	0.00	0.00
Relativity	12.21	16.28	18.02	17.44	**23.84**
Contextual	15.08	21.79	**25.14**	18.43	23.46
Adverb 2	2.65	7.41	**12.70**	5.82	5.29

Table 6.3: Exact match percentages on the new gSCAN Human splits

6.4 Results on the gSCAN Human Dataset

The results for the human data can be seen in Table 6.3. On initial glance, the results seem rather contradictory to the previous section. While in the previous section, the modified baseline with a transformer seemed to perform better across most of the synonym splits, it does not outperform the better multimodal transformer on the gSCAN Human test split. The explanation for this is rather simple. In most of the human data, annotators mostly used the vocabulary present in the original gSCAN dataset to describe the verbs and colors. The most prevalent use of vocabulary outside of the original dataset is present in the descriptions of the shapes. This happens to be the only synonym split in which the multimodal transformer outperforms the rest of the models. Another very common variation is variations in sentence structure, another split in which the multimodal transformers also outperforms all of the baseline variations. People very rarely annotated the examples with commands of an identical structure to the original gSCAN dataset. They often switched around the order of words, or added extraneous details/words that were not entirely necessary. In the sentence structure split, the transformer clearly demonstrated that it is more robust to such changes. While adverb variations were also frequent occurrences in the human dataset, a synonym split the multimodal transformer performed quite poorly on, they did not make enough of a difference in the test split. However, that is not the case in the adverb split, in which we do see that the modified baseline with the transformer does outperform the multimodal transformer. In all of the other splits, the improved performance of the multimodal transformer can be attributed to its

performance on the original splits.

Chapter 7

Conclusion

7.1 Future Work

To fully test generalization to new sentence structures and vocabulary, as well as the generalizations evaluated in the original gSCAN splits, an entirely new dataset should be created. While gSCAN is a great dataset for testing compositional generalizations, it does not lend itself to human annotations. When people are annotating the examples, they often unnaturally described the situation because of the unrealistic environment. Given an environment with more realistic visuals, people would likely write more natural annotations that accurately describe the situation. To solve these issues, an environment could be created that is still a grid world, but has a more realistic 3D visualization that people could rely on for annotation purposes.

There is also a lot of work to be done on the model side. It would be interesting to see if a model with a pretrained multimodal transformer [13][22] (a model pretrained with not just text, but text and image annotations) could make generalizations to entirely new shapes and colors. Such generalizations would be impossible for the models discussed in this work. In the LSTM approach, the world is represented as a matrix which makes such generalizations impossible. In the multimodal transformer approach, our model has only seen the shapes and colors present in the training environment, so such generalizations are also impossible for our multimodal

transformers. However, if a pretrained VilBERT/LXMERT model was used as an encoder, the model could potentially generalize to entirely different shapes and colors because it has seen examples of both in its prior pretraining. Work also needs to be done in order to find models that are just as expressive as they need to be. It is evident that some of the models discussed in this work are overfitting to the training data. Making less expressive models seems to improve some of the generalizations, at the cost of lower performance on the original data. This balance should be explored further, and could likely lead to more capable models.

7.2 Closing Thoughts

Current end-to-end models are unable to correctly execute on the gSCAN Human splits if only trained on the original gSCAN dataset; this sim-to-real transfer is still hard for end-to-end models. While some generalizations are possible individually, once multiple generalization conditions are mixed, models fail to properly execute the given command. Despite the low accuracies on the gSCAN Human splits, the results still demonstrate that all of the models are capable of some form of generalization. We hope that the gSCAN Human dataset will provide researchers with a benchmark for evaluating a model's ability to generalize in an effort to create models with zero-shot capabilities similar to that of humans.

Appendix A

MTurk Instructions

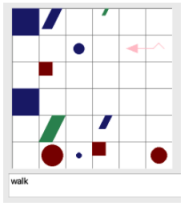
The instructions given to MTurk workers can be seen in the figure on the following page. We made the instructions very specific in an effort to encourage the annotators to keep their sentences somewhat similar to the original dataset. With less specific instructions, annotators often neglected to mention certain features. This was especially true of the different adverbs, which is why an example of all of the different types of movement can be found in the instructions. By including a qualification survey, we also ensured that any workers completing the HITs had thoroughly read through the instructions.

Instructions

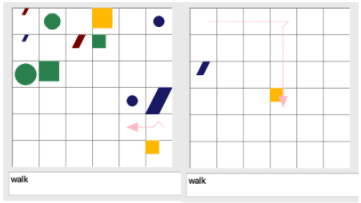
We are researchers from MIT working on creating machines that see and understand the world. To do this we need your help to understand these videos. What command is the robot following? Takes approx. 7 minutes.

In each video, the robot (represented by a pink triangle) finds its way to a target object. In one sentence, write a command to direct the robot's actions. You can give your command any way you like but please note that you **MUST** do the following:

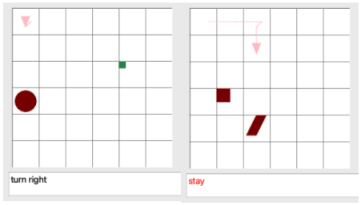
- In one sentence, write the command that the moving pink triangle is following. Always specify the target unambiguously.

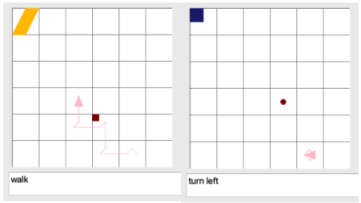


  - For example, in the above Gif don't say "Walk to the red circle." There are two different red circles, so the robot wouldn't know which circle to walk to. Say something like "Walk to the bigger of the two red circles." or "Walk to the red circle next to the blue circle."
- In some cases we will tell you what to talk about or what not to talk about. It's extremely important to follow the instructions on what to mention or not to mention!
- Specify how the robot should move. Usually, the robot goes straight to its target:

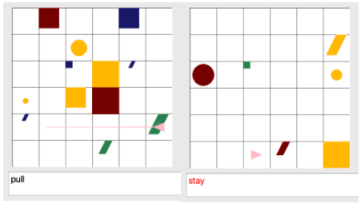


- But the robot can move in other ways as well:





- Finally, specify what the robot does when it reaches its target. Usually, the robot will stop when it reaches its destination. But it can do other things too:



- In your commands, try to talk about the objects, their **sizes**, **shapes**, **colors**, and their relationships. **DO NOT** mention the specific row or column that the object is in.

Figure A-1: The instructions given to each MTurk worker

Bibliography

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2015.

[2] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA, October 2013. Association for Computational Linguistics.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[4] Tong Gao, Qi Huang, and Raymond J. Mooney. Systematic generalization on gscan with language conditioned embedding. In *AACL*, 2020.

[5] Christina Heinze-Deml and Diane Bouchacourt. Think before you act: A simple baseline for compositional generalization. *CoRR*, abs/2009.13962, 2020.

[6] Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. Pretrained transformers improve out-of-distribution robustness. In *ACL*, 2020.

[7] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 11 1997.

[8] Hamid Reza Vaezi Joze, Amirreza Shaban, Michael L Iuzzolino, and Kazuhito Koishida. Mmtm: Multimodal transfer module for cnn fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13289–13299, 2020.

[9] Tristan Karch, Laetitia Teodorescu, Katja Hofmann, Clément Moulin-Frier, and Pierre-Yves Oudeyer. Grounding spatio-temporal language with transformers. In *NeurIPS 2021*, December 2021.

[10] Yen-Ling Kuo, Boris Katz, and Andrei Barbu. Compositional networks enable systematic generalization for grounded language understanding. In *EMNLP*, 2021.

[11] Brenden Lake and Marco Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *International conference on machine learning*, pages 2873–2882. PMLR, 2018.

[12] Chen Liang, Jonathan Berant, Quoc Le, Kenneth D Forbus, and Ni Lao. Neural symbolic machines: Learning semantic parsers on freebase with weak supervision. In *ACL*, 2017.

[13] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[14] Lin Ma, Zhengdong Lu, Lifeng Shang, and Hang Li. Multimodal convolutional neural networks for matching image and sentence. In *Proceedings of the IEEE international conference on computer vision*, pages 2623–2631, 2015.

[15] Alana Marzoev, Samuel Madden, M Frans Kaashoek, Michael Cafarella, and Jacob Andreas. Unnatural language processing: Bridging the gap between synthetic and natural language data. *CoRR*, abs/2004.13645, 2020.

[16] R. Thomas McCoy, Junghyun Min, and Tal Linzen. Berts of a feather do not generalize together: Large variability in generalization across models with similar test set performance. *CoRR*, abs/1911.02969, 2019.

[17] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[18] Linlu Qiu, Hexiang Hu, Bowen Zhang, Peter Shaw, and Fei Sha. Systematic generalization on gSCAN: What is nearly solved and what is next? In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2180–2188, November 2021.

[19] Wasifur Rahman, Md Kamrul Hasan, Sangwu Lee, Amir Zadeh, Chengfeng Mao, Louis-Philippe Morency, and Ehsan Hoque. Integrating multimodal information in large pretrained transformers. In *Proceedings of the conference. Association for Computational Linguistics. Meeting*, volume 2020, page 2359. NIH Public Access, 2020.

[20] Laura Ruis, Jacob Andreas, Marco Baroni, Diane Bouchacourt, and Brenden M Lake. A benchmark for systematic generalization in grounded language understanding. *CoRR*, abs/2003.05161, 2020.

[21] Mike Schuster and Kuldip K. Paliwal. Bidirectional recurrent neural networks. *IEEE Trans. Signal Process.*, 45:2673–2681, 1997.

[22] Hao Tan and Mohit Bansal. LXMERT: learning cross-modality encoder representations from transformers. *CoRR*, abs/1908.07490, 2019.

[23] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[24] Christopher Wang, Candace Ross, Yen-Ling Kuo, Boris Katz, and Andrei Barbu. Learning a natural-language to ltl executable semantic parser for grounded robotics. In *CoRL*, 2020.

[25] Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27, 2015.