CLEVR-Math: A Dataset for Compositional Language, Visual and Mathematical Reasoning

Adam Dahlgren Lindström[1,*], Savitha Sam Abraham[2,*]

[1]Umeå University, Sweden
[2]Örebro University, Sweden

Abstract

We introduce CLEVR-Math, a multi-modal math word problems dataset consisting of simple math word problems involving addition/subtraction, represented partly by a textual description and partly by an image illustrating the scenario. The text describes actions performed on the scene that is depicted in the image. Since the question posed may not be about the scene in the image, but about the state of the scene before or after the actions are applied, the solver envision or imagine the state changes due to these actions. Solving these word problems requires a combination of language, visual and mathematical reasoning. We apply state-of-the-art neural and neuro-symbolic models for visual question answering on CLEVR-Math and empirically evaluate their performances. Our results show how neither method generalise to chains of operations. We discuss the limitations of the two in addressing the task of multi-modal word problem solving.

Keywords

Neuro-Symbolic, Visual Question Answering, Math Word Problem Solving, Multimodal Reasoning

1. Introduction

Math word problems are mathematical problems expressed in natural language. Solving these problems requires one to be able to map the natural language text to a mathematical expression, identifying the known and unknown quantities and the functions to be used. Since this is a task that lies at the intersection of language understanding and reasoning, automatically solving math word problems has been a research topic of interest in Natural Language Processing (NLP) community. Consider the following math word problem,

> **Problem:** *Adam has three apples, and Eve has five. Eve gives Adam all her apples. How many apples does Adam have, if he* eats *one?*
> **Equation:** $X = 3 + 5 - 1$

Minor changes in the text may result in large semantic changes, e.g. changing just one word in the above problem - *eats* to *finds*, will change the equation to $X = 3 + 5 + 1$. Most of the recent efforts in automatic math word problem solving treat it as a translation task (from word

NeSy 2022, 16th International Workshop on Neural-Symbolic Learning and Reasoning, Cumberland Lodge, Windsor, UK
*Corresponding author.
dali@cs.umu.se (A. D. Lindström); savitha.sam-abraham@oru.se (S. S. Abraham)
0000-0002-1112-2981 (A. D. Lindström); 0000-0003-3902-2867 (S. S. Abraham)

problem to equation) and have employed sequence-to-sequence networks or sequence to tree (generating the expression tree of the equation) networks ([1], [2],[3]).

While text-based math word math problems are a great setting for natural language understanding, it would also be interesting to consider word problems which are accompanied by a diagram, and the information required to derive the solution has to be captured from both its textual and visual representations. That is, part of the problem scenario description is expressed as text and the other part is represented in the form of an image. In this paper, we introduce such a multi-modal math word problem dataset, CLEVR-Math (since it is based on CLEVR dataset [4]), where each problem has a textual and a visual description (image). Figure 1 shows an example problem in CLEVR-Math dataset.

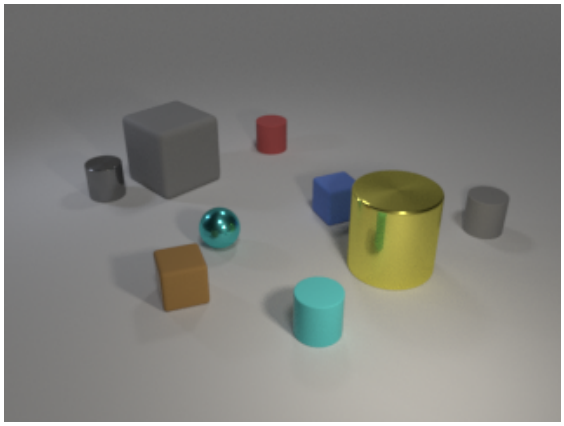


Figure 1.: CLEVR-Math example question *Take away 2 matte cylinders. How many objects are left?* with corresponding mathematical equation $X = 9 - 2$.

While each instance in CLEVR dataset has an image and a natural language query about the scene depicted in the image, in CLEVR-Math, the natural language query may not be about the scene represented in the image, but about the state of the scene after/before a sequence of actions are applied on the scene. The actions in our case are addition/removal of specific type of objects to/from the original scene. We believe this is an interesting problem setting as the ability to envision changes without them being physically manifested is an important aspect of the human mind.

Our contributions are two-fold, we

- construct an open source multi-modal math word problem dataset, CLEVR-Math and
- analyse the performance of state-of-the-art neural and neuro-symbolic (NeSy) solutions for solving such multi-modal problems.

Our results and analysis shows how both neural and NeSy methods are unable to compositionally generalise to chains of operations.

2. Related Work

This section gives an overview of the existing datasets in math word problem solving, multi-modal datasets for visual question answering and existing neural/neuro-symbolic approaches to the tasks that require a combination of visual, language and logical reasoning.

Math Word Problem Solving - Datasets Math Word Problem Solving (MAWPS) [5] was one of the earlier datasets introduced in the domain and collected around 3320 single/multi

equation word problems involving operators $+$, $-$, $*$, $/$. These word problems were annotated with equations involved and the answer (solution of the equation). More recently, larger datasets like Algebra Question Answering with Rationales (AQuA-RAT) [6] were introduced and it has around $100K$ multiple choice questions annotated with equations and a textual explanation for the rationale behind the equations. [7] illustrated the deficiencies in MAWPS dataset by introducing another dataset named Simple Variations on Arithmetic Math word Problems (SVAMP). SVAMP is created by making minor variations to problems in MAWPS. [7] showed that state-of-the-art neural solvers trained on MAWPS performs poorly on the SVAMP dataset.

Visual Question Answering (VQA) and Visual Reasoning - Datasets One of the first VQA datasets proposed was the DAQUAR dataset [8] based on real images of indoor scenes. VQA is another widely used dataset [9] with images from MS-COCO dataset [10]. Questions are manually created and answering these require commonsense knowledge and reasoning. CLEVR dataset [4] is based on automatically generated scenes and questions. Such simulated data gives great control over the distribution of instances. One can decide to generate a training set with images having only a specific combination of objects (red cubes and blue cylinders), and a test set with a different combination of objects (red cylinders and blue cubes), as done in, e.g., CLEVR-Hans [11]. This control allows us to study various aspects like compositional generalisation of systems. Based on these ideas, CLEVR-Math allows us to test the ability of the system to generalise to unseen combinations of actions in the word problem, to e.g. train on single mathematical operations, and test on chains of operations. Closely related is the CLEVRER (Collision Events for Video Representation and Reasoning) dataset [12] and CLEVR-Hyp dataset [13]. The questions on videos in CLEVRER requires reasoning about the state of objects after an video event, instead of after actions in text as in CLEVR-Math. CLEVR-Hyp focus on VQA where reasoning about effects of actions, and CLEVR-Math introduces an additional mathematical reasoning dimension to the problem. GQA is another relevant dataset, where real world images are annotated with rich scene graphs and a large set of relations and attributes, and focuses on compositionality in visual reasoning [14]. Graph learning is a heavily studied area, with applications in multimodal domains such as robotics [15, 16, 17, 18].

Experiments with Kandinsky patterns [19] show that neural networks are easily confounded by visual reasoning tasks with shapes, colors, and patterns that can be difficult to distinguish but follow clear rules. The Winoground dataset [20] shows similar results, where no state-of-the-art visual reasoning method is able to distinguish between two confounding captions and images.

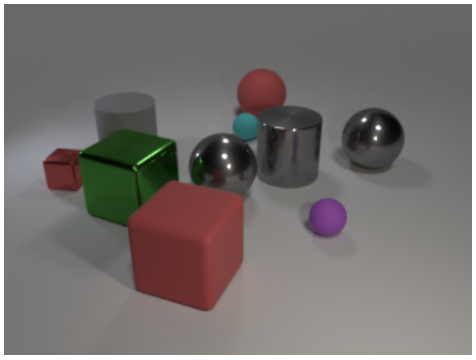
Existing Approaches to VQA Most of the earlier approaches in VQA were based on purely neural models that first encoded the two inputs - the image and the accompanying question into embeddings using networks like Convolutional Neural Networks (CNN) and Long Short Term Memory (LSTM) networks and then the two embeddings were forwarded to a classifier that would then predict the answer to the question ([21], [22]). Another category of approaches are the attention mechanism-based approaches that identified the regions in the image that were relevant to answering the associated question ([23], [24]). Graph neural networks [25] have also been applied in VQA where both text and the image are represented as graphs and a multi-modal vectorial representation is learned that captures the alignment of nodes

in the two graphs. [26] introduced the CLIP models where a representation of the image is learned with natural language supervision by leveraging the already available huge datasets for image captioning. More recently, neuro-symbolic approaches have been used in addressing the task of VQA like Neuro Symbolic Concept Learner (NSCL) [27] and Neuro-Symbolic Visual Question Answering (NS-VQA) [28]. These approaches convert the input image and text into an intermediate semantic representation and then employ a quasi-symbolic program executor to derive an answer from these semantic forms. We use CLIP and NS-VQA as baselines as they are state-of-the-art on multimodal language modelling and on the CLEVR dataset, respectively.

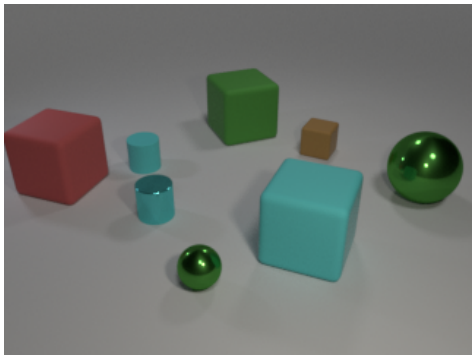
3. CLEVR-Math

We construct the CLEVR-Math dataset as an extension of CLEVR by introducing three new functions and 13 templates. Using the codebase provided with CLEVR, we generate new questions based on the original scenes. We categorise the 13 templates into six types, all based on addition and subtraction. The domain is restricted to numbers between $0 - 10$ to conform with CLEVR.

New CLEVR functions: The three functions that we implement are - `subtraction` and `addition` to perform subtraction and addition, and `choose` to operate on subsets of objects. Instead of removing all blue spheres, `choose` allows us to remove a random number of a specific type of object, e.g. 2 blue spheres out of 4. The random number generated by `choose` replaces a questions "X" placeholder during generation. Figure 2a shows three examples of subtraction, and Figure 2b shows a question requiring multihop reasoning. Appendix A includes more samples from the test set.



(a) (i) *Remove all gray spheres. How many spheres are there? (3)*, (ii) *Take away 3 cubes. How many objects are there?* (7), (iii) *How many blocks must be removed to get 1 block? (2)*



(b) *Take away all large green metallic spheres. Now remove all cyan objects. How many objects are left? (4)*

Figure 2.: Example image-question pairs from CLEVR-Math, 2a showcase addition and subtraction, and 2b shows multihop reasoning. Answers in parenthesis.

Question Categories: The different question categories are shown in Table 1.

- ***Remove group***: All objects belonging to a specific group are removed from the scene.
- ***Insertion***: A specific number of objects are added to the scene.
- ***Count backwards***: The query is about the change - that is the number of objects added/removed from the scene to get a goal state.
- ***Remove subset***: A specific number of objects are removed from the scene.
- ***Adversarial questions***: These are trick questions where the actions may be performed on one object, but the query is about an object that is not affected by the action. The adversarial actions are always on objects that are seen in the image.
- ***Multi-hop***: In contrast to the above questions, multi-hop questions perform sequences of actions (insertion, removal) on the objects. Such questions with chained functions help us test a model's ability to generalise to infinite combinations of operations.

Type	Templates
Remove group	"Remove all <C> <S>s. How many <S>s are there?" "Take away all <Z> <C> <M> <S>s. How many <S>s are there?" "Take away X <C> <S>s. How many objects are there?" "Take away all <C> <S>s. How many objects are there?"
Insertion	"Add X <Z> <C> <M> <S>s. How many <Z> <C> <M> <S>s are here?" "Add X <Z> <C> <M> <S>s. How many objects are there?"
Count backwards	"How many <C> <S>s must be removed to get X <C> <S>s?" "Take away <C> <S>s. How many were removed if there are X <C> <S>s left?"
Multi-hop	"Take away all <Z> <C> <M> <S>s. Remove all <Z2> <C2> <M2> <S2>s. How many objects are left?"
Remove subset	"Remove X <S>s. How many <S>s are there?"
Adversarial questions	"Remove all <C1> <S1>s. Remove all <C2> <S2>s. How many <S1>s are left?" "Remove all <C1> <S1>s. How many <C2> <S2>s are left?"

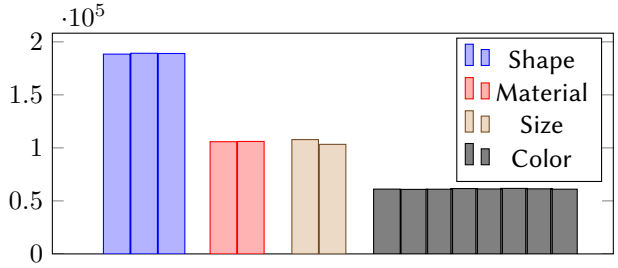
Table 1.
An overview of the different templates implemented by CLEVR-Math. <Z>, <C>, <M>, <S> are instantiated to size, color, material, and shape during the question generation.

Each problem in the dataset is also annotated with it's equivalent functional program based on the CLEVR functions described in the previous section. For example, consider the question from insertion category and it's program (the arguments of an instruction refer to another instruction - indicating it's input is the output of the referred instruction):

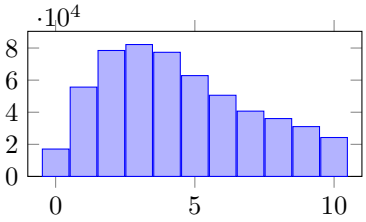
Q: *Add 3 blue cylinders. How many cylinders are there?*
Program: *1. scene, 2. choose[3], 3. count(1), 4. filter_cylinder(1), 5. count(4), addition(2, 5)*

The program contains the choose function - choose[i] operator returns i ($i = 3$ in this case).

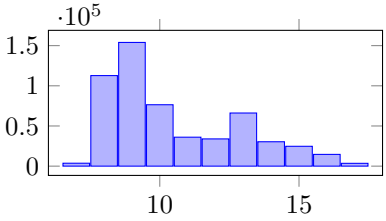
Question generation. To support greater linguistic variation, we add synonyms for addition and subtraction to the template engine. *Subtract* can be replaced with *remove, take away* and *withdraw*, and *addition* with *introduce,* and *insert*. We use the same training and validation scenes as CLEVR, and generate 5000 new scenes as test data. Figure 3 show the distribution of attributes, words, templates and answers in CLEVR-Math, aggregated over the training, validation, and test data. The distribution is reflected in each of the splits.



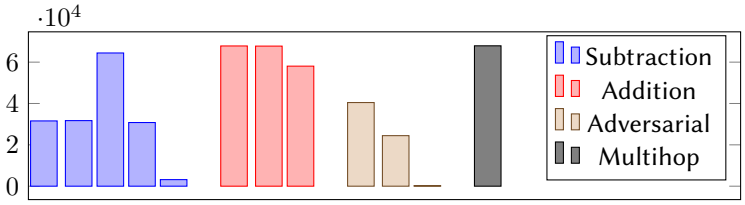
(a) Attribute distribution per category, showing even allocations.



(b) Answer distribution, from 0 to 10.



(c) Distribution of number of words.



(d) Template distribution over categories of templates. Each bar corresponds to a template in each respective category. We see that subset subtraction (i.e., *remove 2 blue cubes*) is underrepresented.

Figure 3.: The attributes are used evenly throughout the dataset, whereas the answers are biased towards the smaller numbers. The numbers are aggregated over all splits.

There are 50 words in the CLEVR-Math vocabulary, where the narrow language puts focus on the mathematical reasoning rather than advanced language capabilities. Figure 3c show that most questions are 8-9 words long, with a second peak at 13 for the multihop questions.

Template	Train	Validation	Test
Subtraction	229364	49149	3281
Addition	193641	41600	2752
Adversarial	65180	13900	950
Multihop	67897	14553	972

Table 2.
Distribution of templates in each data split.

Table 2 shows the distribution of templates, with an approximately equal amount of questions for subtraction and addition, and similarly for adversarial and multihop questions. The ratios are consistent between splits. To test multihop reasoning and compositional generalisability we generate train-validation-test with only singlehop questions in training and validation, and only multihop questions in the test data. Thus, a model using the `CLEVR-Math-multihop` configuration must solve the multihop questions in a zero-shot fashion.

Open sourcing data. We open source CLEVR-Math as a Huggingface dataset [1] with two configurations; `CLEVR-Math` and `CLEVR-Math-multihop`. The extended `CLEVR` source code is available on Github [2]. Table 3 shows the Huggingface dataset card for CLEVR-Math. The template feature allows for filtering to perform, e.g., only singlehop training and multihop testing.

Feature	Type	Example
template	String	`subtraction-multihop`
id	String	`CLEVR_math_test_000010.png`
question	String	*Remove 5 spheres. How many objects are there?*
image	image path	CLEVR_v1.0/images/train/CLEVR_new_000010.png
label	int64, 0-10	5

Table 3.
Huggingface dataset card for CLEVR-Math.

4. Experiments

CLIP [26] is used as a neural baseline. Questions and images are embedded using CLIP, and an additional classification layer is added to predict the correct answer. Fine tuning CLIP on CLEVR-Math as a masked language task before adding classification gave no significant improvements, while consuming significantly more computational resources. CLIP and this classification layer is trained jointly for 10 epochs with early stopping using a batch size of 64.

NS-VQA [28] is used as the neuro-symbolic baseline. Here, a mask-RCNN [29] is trained independently to convert an image to a scene graph. In our experiments, we skip this step and use the actual scene graphs associated with images. The question is parsed into a functional

[1] https://huggingface.co/datasets/dali-does/clevr-math
[2] https://github.com/dali-does/clevr-math

program by a sequence to sequence (Seq2Seq) network based on Bi-LSTM. A quasi-symbolic program executor executes the program generated on the scene graph of the image to return an answer. The Seq2Seq network is pre-trained in a fully supervised fashion by providing it a few examples (around 60 examples) of $(question, program)$ pairs. The pre-trained network is then trained further using REINFORCE algorithm that returns a reward based on whether the program generated could derive the expected answer or not. Supervised pretraining and REINFORCE were run for 1000 and 5000 iterations, respectively, with a batch size of 128. Both CLIP and NS-VQA models were trained on a NVIDIA Tesla P100 GPU computing processor.

Each model is evaluated on each question category, and are trained on 2500, 5000, 10000, and 20000 samples to see the influence of the amount of data. For multihop, training and validation sets with and without multihop questions are used, with the latter named *multihop (0-shot)*.

4.1. Results

Table 4 shows the accuracy of CLIP and NS-VQA on the different categories as well as an aggregated accuracy over the entire dataset. Both the models were trained on $10,000$ samples. NS-VQA performs better than CLIP models for most templates apart from multihop. NS-VQA performs better on subtraction and adversarial problems (both based on 'subtraction' CLEVR function) than addition problems. This could be because the functional programs for addition problems always contain a `choose` operator. It is important to identify the argument to `choose` operator from the problem statement (which is mostly one of the numerical quantities in the word problem) to arrive at the correct answer. Unlike this, there are subtraction and adversarial problems (in remove group) that do not have a `choose` operator in the program. Neither of the

Model	All	Addition	Subtraction	Adversarial	Multihop	Multihop (0-shot)
NS-VQA	0.8840	0.9781	0.9948	0.9957	0.286	0.267
CLIP	0.3464	0.5699	0.3019	0.2848	0.272	0.238

Table 4.
Accuracy on the CLEVR-Math dataset, shown for each template group and aggregated over all templates.

methods perform well on the multi-hop questions, with a clear degradation in the performance for NS-VQA. This is because the question parser of NS-VQA relies on a Seq2Seq network that does not generalize compositionally [30]. CLEVR focus on visual attribute compositionality, and the multihop reasoning introduces higher demands on linguistic compositionality. When multihop questions are included in the training and validation data, naturally both methods improve their performance.

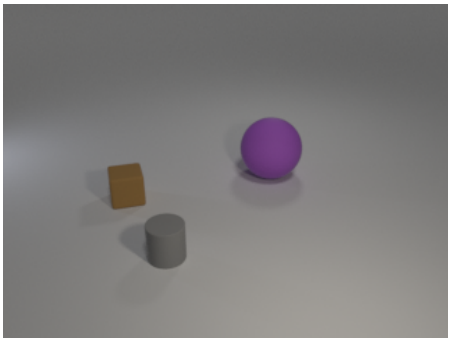
To gain further insight into CLIPs' performance on CLEVR-Math, Appendix C shows a confusion matrix from training CLIP on 20 000 samples and evaluating on all question categories. These results show that most errors made by CLIP is off by ones. This reflects the generative nature of such models, in how they can get the context correct but sometimes miss out on details. We also see how CLIP focus on learning in the range 1-5, reflecting that these problems represent a majority of the problems.

Table 5 shows how different training sizes influence the accuracy. We can see that NS-VQA achieves high accuracy from relatively few examples and plateaus, which is consistent with

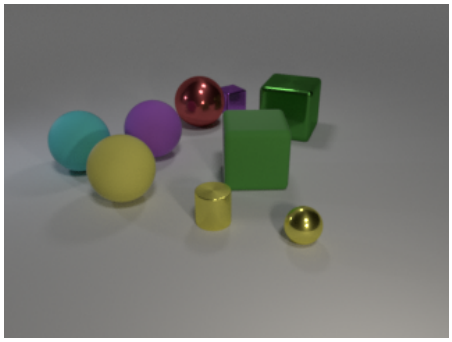
Model	2500	5000	10000	20000
NS-VQA	0.6283	0.8840	0.6795	0.6118
CLIP	0.2918	0.3184	0.3528	0.3464

Table 5.
Accuracy over all templates for different dataset sizes.

the original results on CLEVR. It also seems like NS-VQA is overfitting with more data given, and one hypothesis is that more emphasise is put on the program, but that they are similar enough to confound NS-VQA. In CLEVR, the different questions were more distinguishable from a program perspective. CLIP scales with the number of samples, but plateaus at a much lower accuracy. We note that a larger number of samples could lead to similar performance for CLIP, but at the cost of more computational resources.



(a) *Subtract all small purple matte blocks. Subtract all blocks. How many objects are left?* was answered by CLIP with 3 instead of 2.



(b) *Subtract all red metallic objects. Subtract all yellow objects. How many objects are left?* was answered with 9 instead of 5 by NS-VQA.

Figure 4.: Examples of when CLIP and NS-VQA fails on multihop questions.

We randomly sample 20 correct and 20 incorrect answers from the multihop test data for both CLIP and NS-VQA. Appendix B contains a subset of those samples, and Figure 4 illustrates two incorrect answers. There are no clear patterns of failures, such as only performing one of the actions, but we notice multiple instances where CLIP fails to perform overlapping subtraction, or subtraction when no objects match the description. Another observation is that half of the 20 incorrect answers from CLIP, where on images with only three objects. Scenes with few objects have a much smaller possible action space associated to it, meaning that there is less room for error. In Figure 4a, there are no purple matte blocks to remove, so the corresponding equation is $3 - 0 - 1 = 2$.

5. Conclusions and Future Work

We introduced a new dataset, CLEVR-Math, containing word math problems about visual scenes. Our results show that the state-of-the-art NeSy model, NS-VQA, achieves higher accuracy on

CLEVR-Math with less data and computational resources, than the neural model, CLIP. This is further evidence that neural methods, such as CLIP, are lacking in reasoning capabilities, even after fine tuning. Given that NS-VQA uses perfect scene graphs, the comparison is not completely fair. We still expect the results of learning end-to-end to be consistent with the current results in alignment with the original results on CLEVR for NS-VQA.

CLEVR-Math successfully introduces a focused benchmark for learning and reasoning in multimodal data. There are a few natural extensions to this work, both on further development of the dataset and on evaluation. Extending the benchmark to answers outside of the range 0-10 would provide a more challenging domain, and providing scene graphs for each step of the reasoning chain could open up for other methods. The empirical results show that neither of the models could generalize to chained actions. Hence, it is also of research interest to design neuro-symbolic models where language perception is tackled in a more generalizable manner. Focus should lie on the representations (symbols) that are learned. Other interesting directions is to introduce a representation that is manipulated internally according to the actions as they are read. Adding longer chains of operations, or chains with alternating subtraction and addition, would put even more emphasise on the reasoning capabilities. Finally, there is an opportunity to add confounding information to test the robustness, e.g. by associating each shape with a fixed color during training and randomise it during testing.

6. Acknowledgements

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

References

[1] M.-T. Luong, M. Kayser, C. D. Manning, Deep neural language models for machine translation, in: 19th CoNLL, 2015, pp. 305–309.

[2] Z. Xie, S. Sun, A goal-driven tree-structured neural model for math word problems., in: IJCAI, 2019, pp. 5299–5305.

[3] J. Zhang, L. Wang, R. K.-W. Lee, Y. Bin, Y. Wang, J. Shao, E.-P. Lim, Graph-to-tree learning for solving math word problems, ACL, 2020.

[4] J. Johnson, B. Hariharan, L. Van Der Maaten, L. Fei-Fei, C. Lawrence Zitnick, R. Girshick, Clevr: A diagnostic dataset for compositional language and elementary visual reasoning, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 2901–2910.

[5] R. Koncel-Kedziorski, S. Roy, A. Amini, N. Kushman, H. Hajishirzi, Mawps: A math word problem repository, in: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2016, pp. 1152–1157.

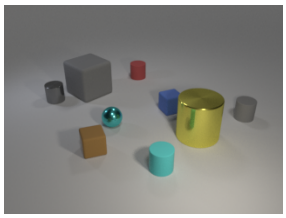
[6] W. Ling, D. Yogatama, C. Dyer, P. Blunsom, Program induction by rationale generation: Learning to solve and explain algebraic word problems, arXiv preprint arXiv:1705.04146 (2017).

[7] A. Patel, S. Bhattamishra, N. Goyal, Are nlp models really able to solve simple math word problems?, arXiv preprint arXiv:2103.07191 (2021).
[8] M. Malinowski, M. Fritz, A multi-world approach to question answering about real-world scenes based on uncertain input, Advances in neural information processing systems 27 (2014).
[9] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, D. Parikh, Vqa: Visual question answering, in: Proceedings of the IEEE international conference on computer vision, 2015, pp. 2425–2433.
[10] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C. L. Zitnick, Microsoft coco: Common objects in context, in: European conference on computer vision, Springer, 2014, pp. 740–755.
[11] W. Stammer, P. Schramowski, K. Kersting, Right for the right concept: Revising neuro-symbolic concepts by interacting with their explanations, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 3619–3629.
[12] K. Yi, C. Gan, Y. Li, P. Kohli, J. Wu, A. Torralba, J. B. Tenenbaum, Clevrer: Collision events for video representation and reasoning, arXiv preprint arXiv:1910.01442 (2019).
[13] S. K. Sampat, A. Kumar, Y. Yang, C. Baral, Clevr_hyp: A challenge dataset and baselines for visual question answering with hypothetical actions over images, arXiv preprint arXiv:2104.05981 (2021).
[14] D. A. Hudson, C. D. Manning, Gqa: A new dataset for real-world visual reasoning and compositional question answering, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 6700–6709.
[15] F. Xia, K. Sun, S. Yu, A. Aziz, L. Wan, S. Pan, H. Liu, Graph learning: A survey, IEEE Transactions on Artificial Intelligence 2 (2021) 109–127. doi:10.1109/TAI.2021.3076021.
[16] F. Yu, J. Tang, W. Yin, Y. Sun, H. Tian, H. Wu, H. Wang, Ernie-vil: Knowledge enhanced vision-language representations through scene graphs, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 35, 2021, pp. 3208–3216.
[17] J. Wald, H. Dhamo, N. Navab, F. Tombari, Learning 3d semantic scene graphs from 3d indoor reconstructions, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 3961–3970.
[18] J. Ji, R. Krishna, L. Fei-Fei, J. C. Niebles, Action genome: Actions as compositions of spatio-temporal scene graphs, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 10236–10247.
[19] A. Holzinger, M. Kickmeier-Rust, H. Müller, Kandinsky patterns as iq-test for machine learning, in: International cross-domain conference for machine learning and knowledge extraction, Springer, 2019, pp. 1–14.
[20] T. Thrush, R. Jiang, M. Bartolo, A. Singh, A. Williams, D. Kiela, C. Ross, Winoground: Probing vision and language models for visio-linguistic compositionality, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 5238–5248.
[21] H. Ben-Younes, R. Cadene, M. Cord, N. Thome, Mutan: Multimodal tucker fusion for visual question answering, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 2612–2620.
[22] A. Fukui, D. H. Park, D. Yang, A. Rohrbach, T. Darrell, M. Rohrbach, Multimodal compact

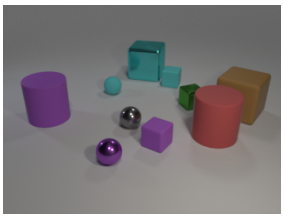
bilinear pooling for visual question answering and visual grounding, arXiv preprint arXiv:1606.01847 (2016).
[23] P. Wang, Q. Wu, C. Shen, A. Dick, A. Van Den Hengel, Fvqa: Fact-based visual question answering, IEEE transactions on pattern analysis and machine intelligence 40 (2017) 2413–2427.
[24] K. J. Shih, S. Singh, D. Hoiem, Where to look: Focus regions for visual question answering, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 4613–4621.
[25] M. Narasimhan, S. Lazebnik, A. Schwing, Out of the box: Reasoning with graph convolution nets for factual visual question answering, Advances in neural information processing systems 31 (2018).
[26] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al., Learning transferable visual models from natural language supervision, in: International Conference on Machine Learning, PMLR, 2021, pp. 8748–8763.
[27] J. Mao, C. Gan, P. Kohli, J. B. Tenenbaum, J. Wu, The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision, arXiv preprint arXiv:1904.12584 (2019).
[28] K. Yi, J. Wu, C. Gan, A. Torralba, P. Kohli, J. Tenenbaum, Neural-symbolic vqa: Disentangling reasoning from vision and language understanding, Advances in neural information processing systems 31 (2018).
[29] K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask r-cnn. corr abs/1703.06870, arXiv preprint arXiv:1703.06870 (2017).
[30] B. Lake, M. Baroni, Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks, in: International conference on machine learning, PMLR, 2018, pp. 2873–2882.

A. Examples

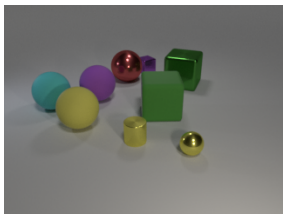
Figure 5c show some examples of questions from different templates in CLEVR-Math.



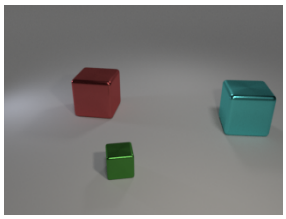
(a) Subtract all gray cylinders. Subtract all gray cubes. How many cylinders are left? (3)



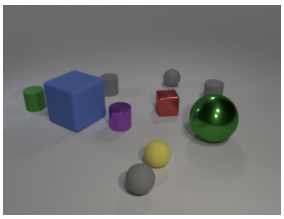
(b) How many cyan cubes must be subtracted to get 1 cyan cubes? (1)



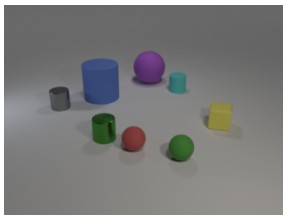
(c) Subtract all purple cylinders. Subtract all yellow blocks. How many cylinders are left? (1)



(d) Add 2 large cubes. How many objects exist? (5)



(e) Subtract 4 cylinders. How many cylinders are left? (0)



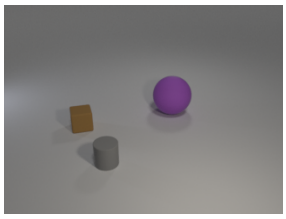
(f) Subtract 0 brown blocks. How many objects are left (8)

Figure 5.: Samples from the test set of CLEVR-Math.

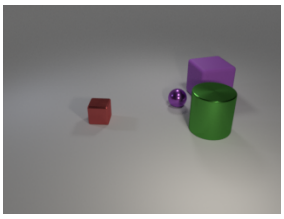
B. Incorrect answers on multihop questions

B.1. CLIP

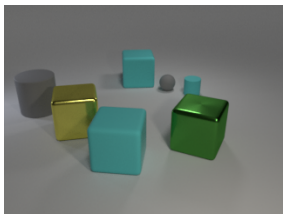
Figure 6 shows random samples of when CLIP fails to answer multihop questions correctly.



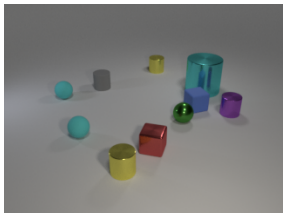
(a) *Subtract all small purple matte blocks. Subtract all blocks. How many objects are left?* was incorrectly answered with 3 instead of 2 by CLIP.



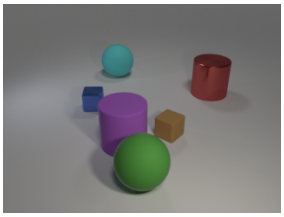
(b) *Subtract all tiny balls. Subtract all big matte blocks. How many objects are left?* was incorrectly answered with 3 instead of 2 by CLIP.



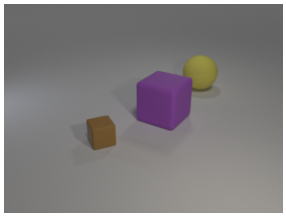
(c) *Subtract all spheres. Subtract all big green shiny cubes. How many objects are left?* was incorrectly answered with 4 instead of 5 by CLIP.



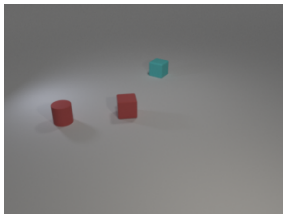
(d) *Subtract all tiny shiny balls. Subtract all purple objects. How many objects are left?* was incorrectly answered with 4 instead of 8 by CLIP.



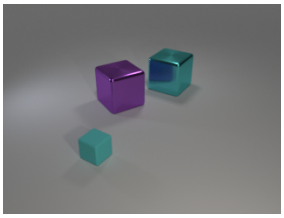
(e) *Subtract all tiny rubber blocks. Subtract all big cyan rubber things. How many objects are left?* was incorrectly answered with 5 instead of 4 by CLIP.



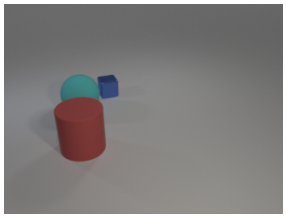
(f) *Subtract all gray balls. Subtract all small cylinders. How many objects are left?* was incorrectly answered with 3 instead of 2 by CLIP.



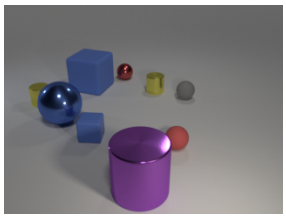
(g) *Subtract all gray balls. Subtract all small cylinders. How many objects are left?* was incorrectly answered with 3 instead of 2 by CLIP.



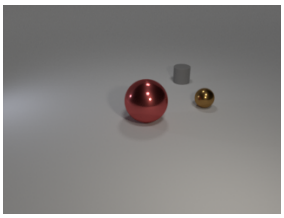
(h) *Subtract all large cyan objects. Subtract all small rubber blocks. How many objects are left?* was incorrectly answered with 3 instead of 1 by CLIP.



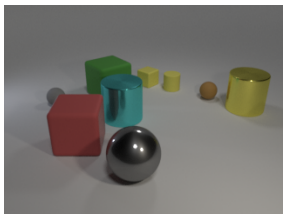
(i) *Subtract all spheres. Subtract all cyan things. How many objects are left?* was incorrectly answered with 3 instead of 2 by CLIP.



(j) *Subtract all big green cylinders. Subtract all big blue cubes. How many objects are left?* was incorrectly answered with 5 instead of 8 by CLIP.



(k) *Subtract all brown things. Subtract all large red spheres. How many objects are left?* was incorrectly answered with 3 instead of 1 by CLIP.

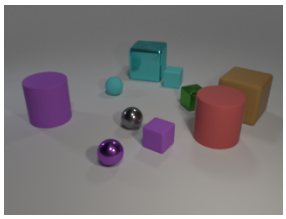


(l) *Subtract all gray objects. Subtract all big green objects. How many objects are left?* was incorrectly answered with 4 instead of 6 by CLIP.

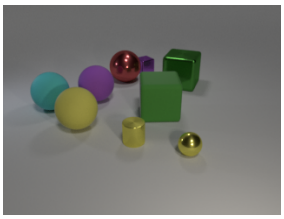
Figure 6.: Sampling of incorrect answers by CLIP on multihop.

B.2. NS-VQA

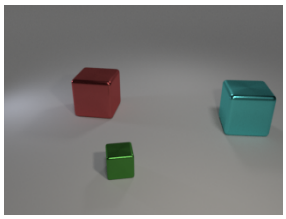
Figure 7 shows random samples of when NS-VQA fails to answer multihop questions correctly.



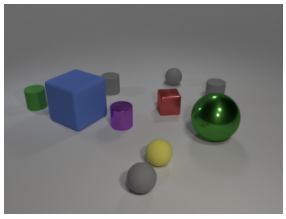
(a) *Subtract all large purple objects. Remove all green metallic objects. How many objects are left?* was incorrectly answered with 7 instead of 8 by NS-VQA.



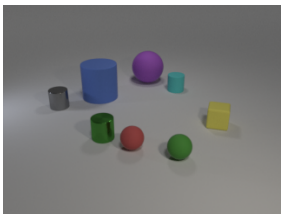
(b) *Subtract all red metallic objects. Subtract all yellow objects. How many objects are left?* was incorrectly answered with 9 instead of 5 by NS-VQA.



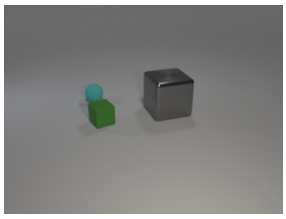
(c) *Subtract all tiny green metallic cubes. Subtract all large brown blocks. How many objects are left?* was incorrectly answered with 0 instead of 2 by NS-VQA.



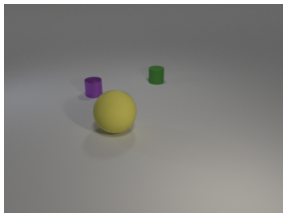
(d) *Subtract all small red objects. Subtract all tiny metal cylinders. How many objects are left?* was incorrectly answered with 6 instead of 8 by NS-VQA.



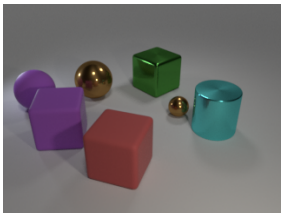
(e) *Subtract all cylinders. Subtract all purple objects. How many objects are left'?* was incorrectly answered with 7 instead of 3 by NS-VQA.



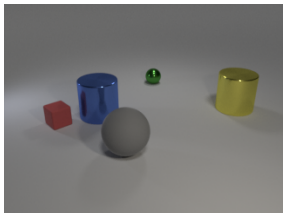
(f) *Subtract all blue metal cylinders. Subtract all gray objects. How many objects are left?* was incorrectly answered with 1 instead of 2 by NS-VQA.



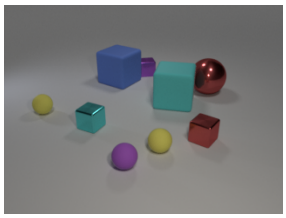
(g) *Subtract all large rubber spheres. Subtract all blue blocks. How many objects are left?* was incorrectly answered with 3 instead of 2 by NS-VQA.



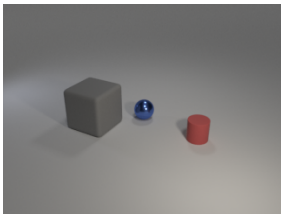
(h) *Subtract all big gray blocks. Subtract all large cylinders. How many objects are left?* was incorrectly answered with 4 instead of 6 by NS-VQA.



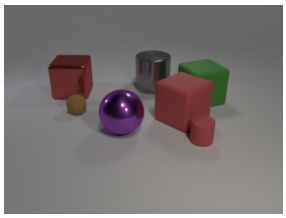
(i) *Subtract all tiny blocks. Subtract all small balls. How many objects are left?* was incorrectly answered with 4 instead of 3 by NS-VQA.



(j) *Subtract all small purple blocks. Subtract all matte objects. How many objects are left?* was incorrectly answered with 7 instead of 3 by NS-VQA.



(k) *Subtract all tiny red rubber objects. Subtract all small blue balls. How many objects are left?* was incorrectly answered with 2 instead of 1 by NS-VQA.



(l) *Subtract all cyan cylinders. Subtract all small rubber objects. How many objects are left?* was incorrectly answered with 6 instead of 5 by NS-VQA.

Figure 7.: Sampling of incorrect answers by NS-VQA on multihop.

C. CLIP confusion matrix

Figure 8 shows a confusion matrix indicating that CLIP is learning something for all labels. It also shows that when an answer is wrong, it is off by one. The confusion matrix also reflects the distribution over answers, showing that most answers are considered by CLIP to lie in the range 1-5.

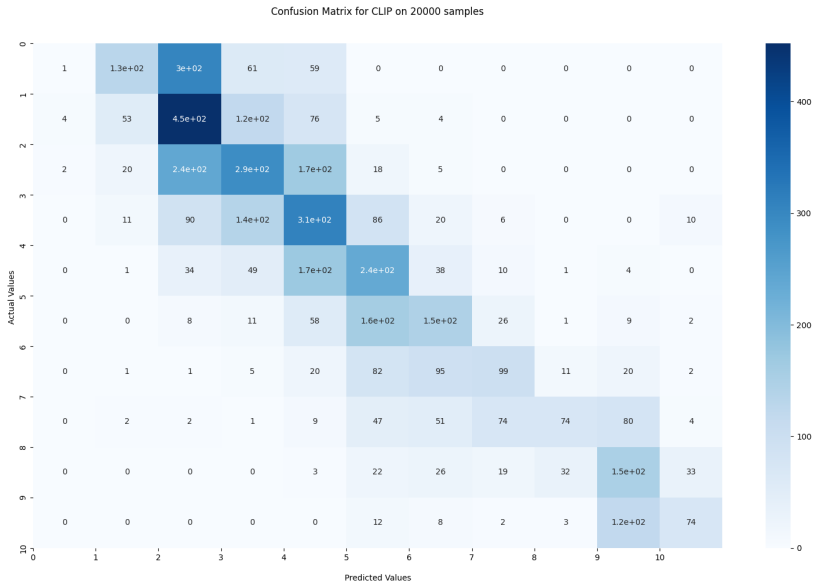


Figure 8.: Confusion matrix for CLIP trained on 20 000 samples.