Neuro-Symbolic Bi-Directional Translation - Deep Learning Explainability for Climate Tipping Point Research

Chace Ashcraft,[1] Jennifer Sleeman,[1] Caroline Tang,[1] Jay Brett,[1] Anand Gnanadesikan[2]

[1] Johns Hopkins University Applied Physics Laboratory
11100 Johns Hopkins Road Laurel, Maryland 20723
[2] Johns Hopkins University
3400 N. Charles St. Baltimore, MD 21218-2683
{chace.ashcraft,jennifer.sleeman,caroline.tang,jay.brett}@jhuapl.edu,gnanades@jhu.edu

Abstract

In recent years, there has been an increase in using deep learning for climate and weather modeling. Though results have been impressive, explainability and interpretability of deep learning models are still a challenge. A third wave of Artificial Intelligence (AI), which includes logic and reasoning, has been described as a way to address these issues. Neuro-symbolic AI is a key component of this integration of logic and reasoning with deep learning. In this work we propose a neuro-symbolic approach called *Neuro-Symbolic Question-Answer Program Translator*, or *NS-QAPT*, to address explainability and interpretability for deep learning climate simulation, applied to climate tipping point discovery. The NS-QAPT method includes a bidirectional encoder-decoder architecture that translates between domain-specific questions and executable programs used to direct the climate simulation, acting as a bridge between climate scientists and deep learning models. We show early compelling results of this translation method and introduce a domain-specific language and associated executable programs for a commonly known tipping point, the collapse of the Atlantic Meridional Overturning Circulation (AMOC).

Introduction

The abundance of climate-related natural disasters (Botzen, Deschenes, and Sanders 2019; Coronese et al. 2019; Jafino et al. 2020), weather extremes (Ebi et al. 2021; Robinson 2021), and poor air quality (Jacob and Winner 2009; Nolte et al. 2018) in recent years has created a sense of urgency in the development of new methodologies for climate research that reduces computation requirements and enables better forecasting, tolerant of a changing environment. Artificial Intelligence (AI) and in particular, deep learning has shown promise in recent years in both data-driven models (Singh et al. 2021) and those which incorporate physical and dynamical properties (Pathak et al. 2022). However, these methods still tend to suffer from poor interpretability and lack of explainability (Garcez and Lamb 2020). In climate and weather related research both of these properties are critical, as often forecasts influence guidance to the general public and policy makers.

As described by Garcez et al.(Garcez and Lamb 2020), the third wave of AI includes deep learning and symbolic representation, described as neuro-symbolic. By incorporating symbolic representation, the black box properties of deep learning models can be informed by a logical understanding of the input and output of the model. Neuro-symbolic refers to the hybridization of symbolic reasoning or computation techniques with deep learning methods. The strengths of deep learning, such as complex pattern recognition and sequence prediction can be augmented by AI methods such as graph-based search and logic systems to produce systems capable of generating robust, human-interpretable predictions that provide a means of explainability.

In this work, we describe a neuro-symbolic model, called *Neuro-Symbolic Question-Answer Program Translator*, or *NS-QAPT*, which symbolically represents deep learning problems and links these representations to natural language. NS-QAPT is a bi-directional translator, that converts natural language questions into surrogate climate model programs and surrogate climate model programs (generated by a deep learning climate simulator) into natural language questions with associated answers. NS-QAPT was designed to bridge the gap between climate tipping point researchers and deep learning models.

To test this methodology, we applied the neuro-symbolic translator and deep learning climate simulator to a known climate problem, the collapse of the Atlantic Meridional Overturning Circulation (AMOC). We demonstrate how NS-QAPT could be integrated with a deep learning climate simulator by using a climate tipping point Generative Adversarial Network (TIP-GAN) (Sleeman et al. 2023b) for climate simulation. Though TIP-GAN could be used to significantly reduce the parameter space by discovering combinations of parameters that lead to models which result in a tipping point, climate researchers need a way to interact with this model and interpret what has been learned. By combining NS-QAPT with TIP-GAN, climate researchers are able to ask natural language questions of what is learned by TIP-GAN, enabling them to potentially direct their own climate research to smaller parameter spaces. We evaluated NS-QAPT using a common neuro-symbolic dataset CLEVR (Johnson et al. 2017) in our previous work (Sleeman et al. 2023a) and in this work we evaluate NS-QAPT with a custom AMOC-specific question program translation language.

Background–The AMOC

The AMOC is a globally circulating current in the Atlantic ocean characterized by warm surface water flowing northward, then cooling, sinking, and flowing back southward. The cooling and increase of salinity of ocean water as it flows northward increases the density of surface water, causing it to sink. It then slowly moves southward along the ocean floor until it can rise in the Pacific and Indian oceans. The northern flow of ocean water from the equator is a significant source of heat energy in the northern hemisphere.

In general, the AMOC plays an important role in the global climate. Small changes to its strength can have potentially global effects, such as significant cooling in the northern hemisphere and changes in precipitation. Some models suggest that the AMOC could weaken or even collapse in the near future (Thornalley et al. 2018; Jackson and Wood 2018), consequences of which may include food insecurity (Benton 2020) and sea level rise (Bakker 2022).

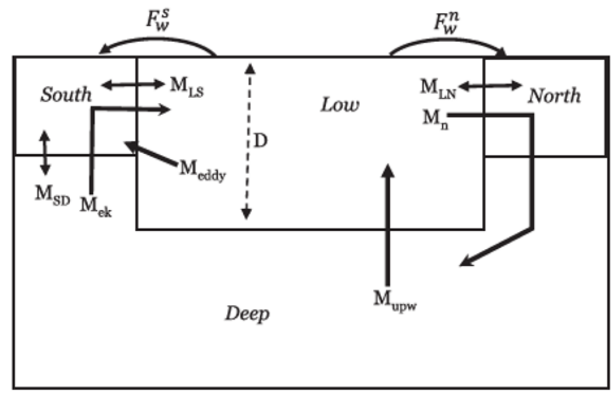

Figure 1: The Gnanadesikan Four Box Model of the AMOC

AMOC Box Models

Large climate systems are sometimes reduced to surrogate models such as a box model (Levermann and Fürst 2010), which simplifies some of the more complex details of the system while maintaining its essential characteristics. This allows the model to theoretically represent the dynamics of their larger counterparts, but are reduced enabling research that would otherwise be computationally infeasible. To experiment with the AMOC and identify states when the AMOC may collapse, we use a four box model from Gnanadesikan et. al. (Gnanadesikan, Kelson, and Sten 2018a), re-implemented in Python[1], as a surrogate for a larger global model. A high-level figure of the box model is shown in Figure 1. *South* and *North* refer to segments of surface water in those latitudes of the Atlantic ocean. The *Low* box similarly represents the surface water in-between, and the *Deep* box represents all deep water flow. M_n refers to the mass transported through the northern box, and is the primary measure of the AMOC's strength. F_w^s and F_w^n are the freshwater fluxes in the southern and northern boxes, respectfully. Due to warming climate, it is possible for these fluxes to grow due to the melting of ice in each region. The influx of freshwater into the ocean affects the salinity of water, potentially perturbing the the whole system. Freshwater flux perturbations are one possible contributor to eventual AMOC collapse.

Related Work

As discussed in (Garcez and Lamb 2020), neuro-symbolic methods are not necessarily new. Khsola and Dillon published a taxonomy of neuro-symbolic approaches along with a neuro-symbolic system called GENUES, which was designed for real-time alarm processing and based some of their previous work (Khosla and Dillon 1993, 1998). Another example of early neuro-symbolic work is (Neagu and Palade 2002), which attempts to fuse artificial neural networks and fuzzy logic. At that time, neural network models generally consisted of shallow, one or two-layer, perceptron models with less than 100 nodes in the hidden layers and perhaps tens of thousands of parameters (Lawrence, Giles, and Tsoi 1998; Canziani, Paszke, and Culurciello 2016; Macukow 2016). AlexNet (Krizhevsky, Sutskever, and Hinton 2012) consists of over 60 million parameters. Continued advancements both in neural network architecture design and faster compute have been integral to the success of deep learning. Early attempts at applying to neuro-symbolic techniques to climate include forecasting red tides (Fdez-Riverola and Corchado 2003) and energy management (Velik, Zucker, and Dietrich 2011; Velik 2013). Fdez-Riverola uses neural networks to index into a case-based reasoner, which stores latent representations of previous fuzzy logic rules that predicted accurate phytoplankton concentrations in the past. Velik uses neural networks to help covert raw sensor information into aggregate latent information, which is then used by a rule-based planning and control module to regulate the power consumption of the electrical devices in a home.

While recent applications of deep learning to climate are becoming plentiful (Rasp, Pritchard, and Gentine 2018; Reichstein et al. 2019; Bury et al. 2021; Singh et al. 2021; Schultz et al. 2021), and climate related neuro-symbolic work with shallow neural networks exists, we were not able to find recent applications of neuro-symbolic (i.e. that use deep learning) methods applied to climate change–and, more specifically, to climate tipping points–in our literature survey.

Model Design

NS-QAPT is inspired by the neuro-symbolic Concept Learner (NS-CL) by Mao et. al. (Mao et al. 2019) and CLEVRER (Yi et al. 2019). The NS-CL learns to associate latent representations of objects in a scene with given concept words from a domain specific language (DSL), as well as learning to manipulate and execute quasi-symbolic programs to answer questions about the scene. The authors leverage the CLEVR dataset (Johnson et al. 2017), which consists of sets of computer generated images of objects, questions about the relationships between objects in the images, and corresponding "programs" that answer the given questions. First, the NS-CL uses a perception network to extract latent representations of objects in an image. Then the extracted representations are given to a reasoning module,

[1] https://github.com/JHUAPL/PACMANs

which identifies the concepts represented in the latent representations and what operations to perform on each concept to answer the question. The reasoning operations consist of implementations of the programs given in the CLEVR dataset. Training is done end-to-end using stochastic gradient decent and the REINFORCE (Williams 1992) algorithm.

NS-QAPT differs from NS-CL (Mao et al. 2019) or CLEVRER (Yi et al. 2019) in that it does not require a perception module to extract concepts from an image since this is a text-only problem. NS-QAPT is also a bidirectional translation between natural language and programs, which is not part of NS-CL (Mao et al. 2019) or CLEVRER (Yi et al. 2019) methodology. NS-QAPT learns programs in a purely sequence-to-sequence manner, rather than from search or reinforcement learning.

The bidirectional question-to-program translation is accomplished via a triangular shaped system of model architectures as seen in Figure 2. The three pieces of NS-QAPT include a question-to-question (QTQ) auto-encoder, a question-to-program (QTP) encoder-decoder, and a program-to-question (PTQ) encoder-decoder. All parts share the same latent space and token embedding, and are optimized jointly during training.

Let B denote a batch of N examples $\mathbf{x}_i = (x_i^Q, x_i^P)$, where x_i^Q is the vector of integer tokens representing the i^{th} natural language question in the batch, and x_i^P is the vector representing the corresponding tokenized program. We denote NS-QAPT's predicted output of $\mathbf{x}_i$ as $\hat{\mathbf{y}}_i = (\hat{y}_i^{QTQ}, \hat{y}_i^{QTP}, \hat{y}_i^{PTQ})$, where $\hat{y}_i^{QTQ}$ is NS-QAPT's question-to-question auto-encoder prediction, $\hat{y}_i^{QTP}$ its question-to-program encoder-decoder prediction, and $\hat{y}_i^{PTQ}$ its program-to-question encoder-decoder prediction. Then ground truth may be written as

$$\mathbf{y}_i = (y_i^{QTQ}, y_i^{QTP}, y_i^{PTQ}) = (x_i^Q, x_i^P, x_i^Q).$$

Let L_{CE} denote the standard cross-entropy loss, we define the **total cross-entropy**, L_{TCE}, as

$$\begin{aligned} L_{TCE}(\hat{\mathbf{y}}, \mathbf{y}) =& L_{CE}(\hat{y}^{QTQ}, y^{QTQ}) \\ &+ L_{CE}(\hat{y}^{QTP}, y^{QTP}) \\ &+ L_{CE}(\hat{y}^{PTQ}, y^{PTQ}). \end{aligned}$$

Let $|v|$ be the length of a vector v, and $\Big|z\Big|$, be the absolute value of a scalar z. We define the **total length difference**, L_{TLD}, as

$$\begin{aligned} L_{TLD}(\hat{\mathbf{y}}, \mathbf{y}) =& \Big||\hat{y}^{QTQ}| - |y^{QTQ}|\Big| \\ &+ \Big||\hat{y}^{QTP}| - |y^{QTP}|\Big| \\ &+ \Big||\hat{y}^{PTQ}| - |y^{PTQ}|\Big|. \end{aligned}$$

Finally, let α be a constant scalar. Using this notation, we may write the loss function, $\mathcal{L}$, for NS-QAPT as follows:

$$\mathcal{L}(B) = \frac{1}{N}\sum_i^N L_{TCE}(\hat{\mathbf{y}}_i, \mathbf{y}_i) - \alpha L_{TLD}(\hat{\mathbf{y}}_i, \mathbf{y}_i)$$

where $\alpha = 0.001$. In the cases where $L_{TLD}(\hat{\mathbf{y}}, \mathbf{y}) > 0$, the inputs to L_{TCE} are truncated to the length of the shorter of the predicted and ground truth vectors.

At a high level, the loss on a batch consists of summing the cross-entropy between model predictions and ground truth for QTQ, QTP, and PTQ data, subtracting the absolute values of the differences in sequence lengths between predictions and ground truth, scaled down by a constant factor, and then returning the mean over all N examples in the batch.

NS-QAPT's question and program encoders share a sequence representation consisting of a word embedding of size 512 and a modified learned positional embedding (Mao et al. 2019) of size 128. Both encoders are bidirectional, 2-layer Gated Recurrent Units (GRUs) (Cho et al. 2014) with a hidden size of 512. The decoders have a similar architectures, each being a single-direction, single-layer GRU with hidden size 1024 followed by two linear layers. The first layer is size 512 and followed by a LeakyReLU (Maas et al. 2013), and the second is size 253, which is the size of the vocabulary. The vocabulary was constructed from tokens in the CLEVR dataset and tokens from the AMOC questions (For the CLEVR experiment, the vocabulary only consisted of CLEVR tokens, and thus these layers were smaller for our CLEVR experiment.). Currently, numerical values are converted to "`VALUE`" tokens pre-encoding and are stored in a dictionary to be passed to the decoders. `VALUE` tokens are replaced with numerical values post-decoding in the same order they were encoded. While dealing with numerical values this way works reasonably well, we hope future work will use more advanced methods, such as a learned association with numerical values and their position in the sequence.

AMOC Dataset

We evalauted our method using two question-answer-program translation datasets. The first was based on the CLEVR (Johnson et al. 2017) dataset, which a well-developed dataset about relationships of geometric objects in images used to benchmark neuro-symbolic methods. We described the results of evaluating our methodology using the CLEVR dataset in our previous work (Sleeman et al. 2023a). The second is based on AMOC-related questions and answers, and program translations, created specifically for this work.

We generated a custom set of AMOC-collapse questions and their corresponding programs to further evaluate our model. Our approach is to define question "forms" in which words and numerical values may be inserted to create valid questions answerable by a set of implemented programs. For example, one question form is "What is the value of M_n at time step $\{1\}$ if $\{2\}$ is $\{3\}$?" where *M_n* represents the mass transported through the northern box (M_n), and $\{1\}$, $\{2\}$, and $\{3\}$ are placeholders for possible values. Replacing $\{1\}$

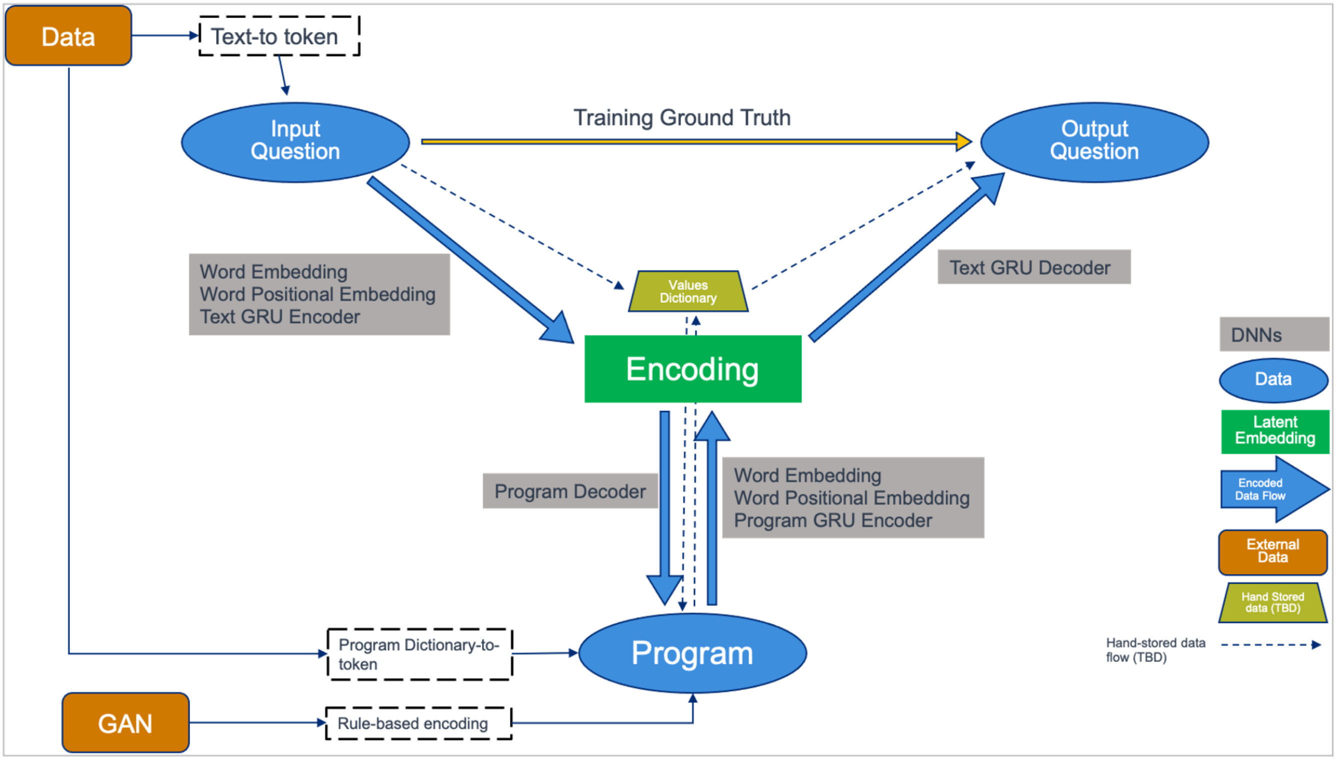

Figure 2: NS-QAPT's bidirectional text-to-program translation architecture. NS-QAPT is a combination of an auto-encoder and two encoder-decoder models. All encoders and decoders share a latent space.

with 4000, {2} with Fwn, and {3} with 5000 results in the question:

> "What is the value of M_n at time step 4000 if Fwn is 5000?"

The corresponding program is:

"FinalValue(
four_box_model(
SetTo(N,4000),SetTo(Fwn,5000)),
M_n)"

where "`four_box_model`" runs the Four Box Model simulation for a vector of parameters, "`SetTo`(x, y)" sets the Four Box Model parameter x to y, and "`FinalValue`(V, z)" extracts the data representing the variable z from V, the set of outputs of the Four Box Model simulation, and returns value of that data at the final step of the simulation.

Each question has a discrete set of parameters from which question, program pairs may be generated. If a question requires a numerical entry, a value is generated by adding noise to the default box-model value for the associated parameter, or from a standard normal distribution if the entry does not have a related default box-model value. Noise is also constrained to ensure each value is within a reasonable range for its a parameter.

Some questions allow for repeated phrases using different words or numbers, creating a combinatorial expansion of similar questions. For example, "If I set Fwn to 5.8e4, M_ek to 2.6e7, will M_n increase?" could be extended to "If I set Fwn to 5.8e4, M_ek to 2.6e7, and D_low0 to 439, will M_n increase?" essentially adding another `SetTo` call to the `four_box_model` function. Each `SetTo` call (or clause) can be given in any order, and for as many parameters as desired, creating the combinatoral expansion of questions. For our dataset, we use no more than three parameters per question.

Finally, some phrases may also be substituted with different, synonymous phrases, in order to build more diversity into dataset. For example "If I set Fwn to" may be replaced with "Setting the freshwater flux in the northern ocean to." without changing the meaning of the question and thus not changing the corresponding program. We include several such replacements to generate our current dataset.

Table 1 shows some example questions from the dataset, with Table 2 showing their corresponding programs.

Experimental Setup and Results

We describe the experimental setup when applying this methodology to the AMOC question dataset. The metric we report is the normalized Levenshtein distance (Yujian and Bo 2007). Levenshtein distance is a distance metric for the number of replacements required to change one sequence into another. Normalized Levenshtein distance converts this measure to be in the interval $[0, 100]$, and like accuracy, greater values mean more similar sequences. Therefore approaching 100 is the desired behavior.

Example
What is the value of M_n at time step 4000 if Fwn is 5000?
If Fwn is 45113 and M_ek is 2.7e7, does the AMOC collapse?
What is the final value of the AMOC when Fwn is 49243?
Does Fwn collapse the AMOC at 49483?
If I set Fwn to 5.8e4, M_ek to 2.6e7, and D_low0 to 439, will M_n increase?
If I increase Fwn by 2052, will M_n increase?
If I increase Fwn by 720, will salinity in the northern box increase?

Table 1: Examples of question types generated for our AMOC-collapse dataset. Variables from the Four Box model (Gnanadesikan, Kelson, and Sten 2018b): *M_n* (mass transport through the northern box), *Fwn* (freshwater flux in the northern box), *M_ek* (Ekman transport), and *D_low0* (start depth of the low box).

Example
FinalValue(four_box_model(SetTo(N,4000),SetTo(Fwn,5000)), M_n)
ChangeSign(four_box_model(SetTo(Fwn,45113),SetTo(M_ek,2.7e7)), M_n)
FinalValue(four_box_model(SetTo(Fwn,49243)),M_n)
ChangeSign(four_box_model(SetTo(Fwn,49483)),M_n)
IncreaseOf(four_box_model(SetTo(Fwn, 5.8e4), SetTo(M_ek, 2.6e7), SetTo(D_low0, 439)), M_n)
IncreaseOf(four_box_model(IncreaseBy(Fwn,2052)),M_n)
IncreaseOf(four_box_model(IncreaseBy(Fwn,720)),S_north)

Table 2: Examples of programs generated for our AMOC-collapse dataset. The *four_box_model* function represents a run of the Four Box model (Gnanadesikan, Kelson, and Sten 2018b). Arguments are model parameters to update from the defaults. We assume the output of this model is AMOC time-series data, including north box mass transport *M_n*. *S_north* represents the salinity of the water in the northern box (see Table 1 for information on other Four Box model variables).

AMOC

We train NS-QAPT for three epochs on a dataset consisting of 250,000 examples, balanced to have approximately equal numbers of each question and equal numbers of question sequence lengths. We then test on a holdout set of 25,000 examples, which are separated from the training data prior to balancing, and does not contain repeat examples like in the training data in order to maintain balance. We perform this experiment three times with different seeds, and refer to each simply as Experiment 1, Experiment 2, and Experiment 3. As seen in Figure 3, all models converge within the first epoch (∼4000 steps).

In Table 3, we also see that the performance between the models is also very similar.

Analysis

Figure 4 shows the distribution of normalized Levenshtein distances over all examples in the test dataset. We can see that the QTQ and QTP CDFs are completely concentrated over the 95-100 distances, while PTQ is more spread out over distances from 20 to 100. The overall distribution helps show the contribution of PTQ performance to the overall normalized Levenshtein distance.

In general, we see that the model performs quite well on the QTQ and QTP portions of evaluation, but struggles with PTQ. For QTQ and QTP, the model appears to reproduce ground truth with almost 100% accuracy for all the different types of questions, but the normalized Levenshtein distance for PTQ is closer to 70 (100 is perfect). Closer examination shows that this may, in part, be a consequence of the lack of variety in the set of training programs. 23,502 of the 25,000 test questions all came from the same question form. Indeed, as seen in Table 4, the consequence of the data generation and train and test split is a very small number of test examples for most questions.

Figure 5, shows test performance by question for Experiment 1. The NS-QAPT model performs well on questions 1-7, nearing 100 normalized Levenshtein distance, but drops to between 60% and 80% for questions 8-10. As these questions constitute more than 99% of the dataset, the PTQ performance in Table 3 is close to the performance on just question 8. Taking an unweighted mean of the performance over questions gives a mean normalized Levenshtein distance of **88.8**, which is significantly higher.

Questions 8-10 have so many more examples because they their forms allow for multiple calls to the `SetTo` method, and have several synonymous phrases that many interchanged to produce variants of the same question. However, while this is helpful for producing many questions with subtle variations, the many of these variations produce the same program. The lack of variety in the program set may be the cause of poorer performance on these questions.

It is also worth noting that the normalized Levenshtein distance measure does not capture correctness of meaning between predictions and ground truth. For example, consider the following prediction vs versus ground truth for both CLEVR and AMOC PTQ outputs:

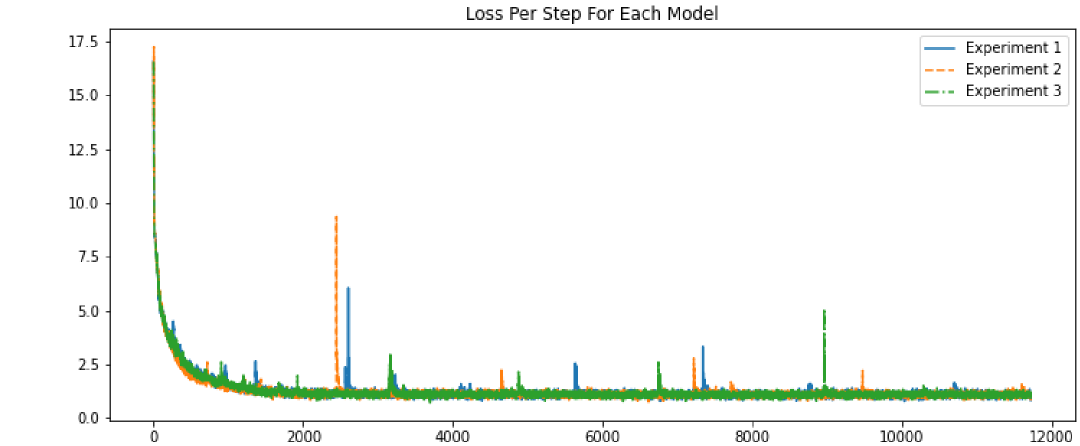

Figure 3: AMOC mean loss per optimization step between three experiments training the translation model.

Model	QTQ	QTP	PTQ
1	99.99	99.99	61.69
2	99.99	99.99	61.16
3	99.99	99.99	63.04
Mean	99.99	99.99	61.97
StDev	0.0006	0.0002	0.79

Table 3: AMOC mean evaluation scores between the three NS-QAPT models. Normalized Levenshtein distance (Yujian and Bo 2007) is a normalized measure of the number of replacements required to convert the predicted sequence to the ground truth sequence. NS-QAPT shows low variance between training runs.

Q1	Q2	Q3	Q4	Q5
4	26	3	4	67

Q6	Q7	Q8	Q9	Q10
4	4	23,502	198	1,188

Table 4: Resulting question counts for the test set after 90/10 train-test split.

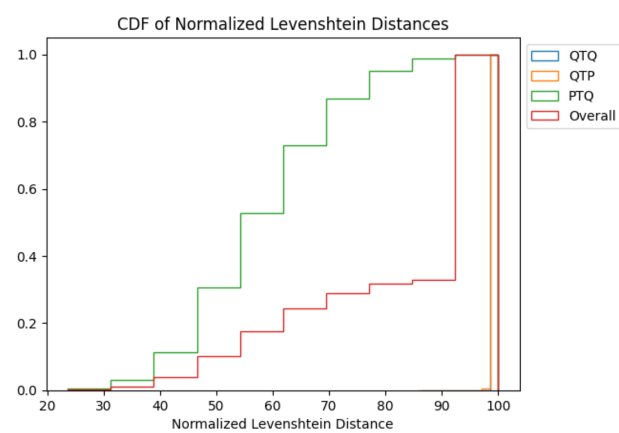

Figure 4: CDF plot of normalized Levenshtein distance performance on AMOC questions for Experiment 1 NS-QAPT models.

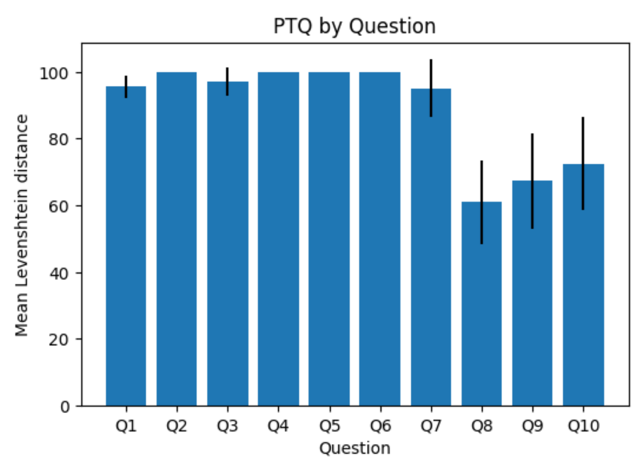

Figure 5: Mean and standard deviation normalized Levenshtein distance by question for Experiment 1 PTQ performance.

Prediction: "**if i increase epsilon by 4.24e-06, will temperature in the low latitude box increase?**"

Ground Truth: "**by increasing epsilon by 4.24e-06, will temperature in the low latitude box increase?**"

Levenshtein distance **93.4**

Examining various model predictions shows that common mistakes include missing and repeated tokens, in addition to semantically related errors such as synonym substitutions.

Conclusions and Future Work

Neuro-symbolic methods have the potential to overcome reluctance in using deep learning models for weather and climate forecasting, as they provide a means to interrogate what is learned by the neural methods and a natural language for easy adoption. By coupling the neuro-symbolic method with a deep learning simulator, these methods can work together to reduce the search spaces that are required for climate modeling problems such as AMOC collapse or other types of climate tipping points, potentially enabling faster and more accurate forecasting, with results that are interpretable and explainable.

We described a neuro-symbolic bi-directional translation model to translate between questions and programs that pertain to a neural simulation built to identify areas in state space that warrant climate modeling exploration as it relates to AMOC collapse. We introduced an AMOC question dataset, and showed how our model is able to translate from questions to programs with a high degree of accuracy and translate from programs to questions with slightly lower accuracy. As we advance the AMOC language further we expect to enable a richer set of questions and improve the program-to-question performance. Future work will also include exploring semantic methods to support one-to-many translations from programs to question.

References

Bakker, P. 2022. Ocean sensitivity to freshwater. *Nature Climate Change*, 12(5): 419–420.

Benton, T. G. 2020. Running AMOC in the farming economy. *Nature Food*, 1(1): 22–23.

Botzen, W. W.; Deschenes, O.; and Sanders, M. 2019. The economic impacts of natural disasters: A review of models and empirical studies. *Review of Environmental Economics and Policy*.

Bury, T. M.; Sujith, R.; Pavithran, I.; Scheffer, M.; Lenton, T. M.; Anand, M.; and Bauch, C. T. 2021. Deep learning for early warning signals of tipping points. *Proceedings of the National Academy of Sciences*, 118(39): e2106140118.

Canziani, A.; Paszke, A.; and Culurciello, E. 2016. An Analysis of Deep Neural Network Models for Practical Applications.

Cho, K.; Van Merriënboer, B.; Bahdanau, D.; and Bengio, Y. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*.

Coronese, M.; Lamperti, F.; Keller, K.; Chiaromonte, F.; and Roventini, A. 2019. Evidence for sharp increase in the economic damages of extreme natural disasters. *Proceedings of the National Academy of Sciences*, 116(43): 21450–21455.

Ebi, K. L.; Vanos, J.; Baldwin, J. W.; Bell, J. E.; Hondula, D. M.; Errett, N. A.; Hayes, K.; Reid, C. E.; Saha, S.; Spector, J.; and Berry, P. 2021. Extreme Weather and Climate Change: Population Health and Health System Implications. *Annual Review of Public Health*, 42(1): 293–315. PMID: 33406378.

Fdez-Riverola, F.; and Corchado, J. M. 2003. Forecasting red tides using an hybrid neuro-symbolic system. *AI Communications*, 16(4): 221–233.

Garcez, A. d.; and Lamb, L. C. 2020. Neurosymbolic AI: the 3rd wave. *arXiv preprint arXiv:2012.05876*.

Gnanadesikan, A.; Kelson, R.; and Sten, M. 2018a. Flux correction and overturning stability: Insights from a dynamical box model. *Journal of Climate*, 31(22): 9335–9350.

Gnanadesikan, A.; Kelson, R.; and Sten, M. 2018b. Flux correction and overturning stability: Insights from a dynamical box model. *Journal of Climate*, 31(22): 9335–9350.

Jackson, L.; and Wood, R. 2018. Hysteresis and resilience of the AMOC in an eddy-permitting GCM. *Geophysical Research Letters*, 45(16): 8547–8556.

Jacob, D. J.; and Winner, D. A. 2009. Effect of climate change on air quality. *Atmospheric Environment*, 43(1): 51–63. Atmospheric Environment - Fifty Years of Endeavour.

Jafino, B. A.; Walsh, B.; Rozenberg, J.; and Hallegatte, S. 2020. Revised estimates of the impact of climate change on extreme poverty by 2030.

Johnson, J.; Hariharan, B.; van der Maaten, L.; Fei-Fei, L.; Zitnick, C. L.; and Girshick, R. 2017. CLEVR: A Diagnostic Dataset for Compositional Language and Elementary Visual Reasoning. In *CVPR*.

Khosla, R.; and Dillon, T. 1993. Combined Symbolic-Artificial Neural Net Alarm Processing System. In *11th Power Systems Computation Conference*, 259–266.

Khosla, R.; and Dillon, T. 1998. Welding symbolic AI systems with neural networks and their applications. In *1998 IEEE International Joint Conference on Neural Networks Proceedings. IEEE World Congress on Computational Intelligence (Cat. No. 98CH36227)*, volume 1, 29–34. IEEE.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In Pereira, F.; Burges, C.; Bottou, L.; and Weinberger, K., eds., *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc.

Lawrence, S.; Giles, C. L.; and Tsoi, A. C. 1998. What size neural network gives optimal generalization? Convergence properties of backpropagation. Technical report.

Levermann, A.; and Fürst, J. J. 2010. Atlantic pycnocline theory scrutinized using a coupled climate model. *Geophysical research letters*, 37(14).

Maas, A. L.; Hannun, A. Y.; Ng, A. Y.; et al. 2013. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, 3. Atlanta, Georgia, USA.

Macukow, B. 2016. Neural Networks – State of Art, Brief History, Basic Models and Architecture. In Saeed, K.; and Homenda, W., eds., *Computer Information Systems and Industrial Management*, 3–14. Cham: Springer International Publishing. ISBN 978-3-319-45378-1.

Mao, J.; Gan, C.; Kohli, P.; Tenenbaum, J. B.; and Wu, J. 2019. The Neuro-Symbolic Concept Learner: Interpreting Scenes, Words, and Sentences From Natural Supervision.

Neagu, C.-D.; and Palade, V. 2002. Modular Neuro-Fuzzy Networks Used in Explicit and Implicit Knowledge Integration. In *FLAIRS Conference*, 277–281.

Nolte, C. G.; Spero, T. L.; Bowden, J. H.; Mallard, M. S.; and Dolwick, P. D. 2018. The potential effects of climate change on air quality across the conterminous US at 2030 under three Representative Concentration Pathways. *Atmospheric chemistry and physics*, 18(20): 15471–15489.

Pathak, J.; Subramanian, S.; Harrington, P.; Raja, S.; Chattopadhyay, A.; Mardani, M.; Kurth, T.; Hall, D.; Li, Z.; Azizzadenesheli, K.; Hassanzadeh, P.; Kashinath, K.; and Anandkumar, A. 2022. FourCastNet: A Global Data-driven High-resolution Weather Model using Adaptive Fourier Neural Operators.

Rasp, S.; Pritchard, M. S.; and Gentine, P. 2018. Deep learning to represent subgrid processes in climate models. *Proceedings of the National Academy of Sciences*, 115(39): 9684–9689.

Reichstein, M.; Camps-Valls, G.; Stevens, B.; Jung, M.; Denzler, J.; Carvalhais, N.; et al. 2019. Deep learning and process understanding for data-driven Earth system science. *Nature*, 566(7743): 195–204.

Robinson, W. A. 2021. Climate change and extreme weather: A review focusing on the continental United States. *Journal of the Air & Waste Management Association*, 71(10): 1186–1209. PMID: 34128774.

Schultz, M. G.; Betancourt, C.; Gong, B.; Kleinert, F.; Langguth, M.; Leufen, L. H.; Mozaffari, A.; and Stadtler, S. 2021. Can deep learning beat numerical weather prediction? *Philosophical Transactions of the Royal Society A*, 379(2194): 20200097.

Singh, M.; Kumar, B.; Rao, S.; Gill, S. S.; Chattopadhyay, R.; Nanjundiah, R. S.; and Niyogi, D. 2021. Deep learning for improved global precipitation in numerical weather prediction systems. *arXiv preprint arXiv:2106.12045*.

Sleeman, J.; Chung, D.; Ashcraft, C.; Brett, J.; Gnanadesikan, A.; Kevrekidis, Y.; Hughes, M.; Haine, T.; Pradal, M.-A.; Gelderloos, R.; Tang, C.; Saksena, A.; and White, L. 2023a. Using Artificial Intelligence to aid Scientific Discovery of Climate Tipping Points.

Sleeman, J.; Chung, D.; Gnanadesikan, A.; Brett, J.; Kevrekidis, Y.; Hughes, M.; Haine, T.; Pradal, M.-A.; Gelderloos, R.; Ashcraft, C.; Tang, C.; Saksena, A.; and White, L. 2023b. A Generative Adversarial Network for Climate Tipping Point Discovery (TIP-GAN).

Thornalley, D. J.; Oppo, D. W.; Ortega, P.; Robson, J. I.; Brierley, C. M.; Davis, R.; Hall, I. R.; Moffa-Sanchez, P.; Rose, N. L.; Spooner, P. T.; et al. 2018. Anomalously weak Labrador Sea convection and Atlantic overturning during the past 150 years. *Nature*, 556(7700): 227–230.

Velik, R. 2013. Cognitive Architectures as Building Energy Management System for Future Renewable Energy Scenarios. *International Journal of Computational Science and Engineering Investigations*, 2: 68–72.

Velik, R.; Zucker, G.; and Dietrich, D. 2011. Towards automation 2.0: a neurocognitive model for environment recognition, decision-making, and action execution. *EURASIP Journal on Embedded Systems*, 2011: 1–11.

Williams, R. J. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Reinforcement learning*, 5–32.

Yi, K.; Gan, C.; Li, Y.; Kohli, P.; Wu, J.; Torralba, A.; and Tenenbaum, J. B. 2019. Clevrer: Collision events for video representation and reasoning. *arXiv preprint arXiv:1910.01442*.

Yujian, L.; and Bo, L. 2007. A normalized Levenshtein distance metric. *IEEE transactions on pattern analysis and machine intelligence*, 29(6): 1091–1095.

Acknowledgments

Approved for public release; distribution is unlimited. This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Agreement No. HR00112290032.