AtmoRep

Large scale representation learning of atmospheric dynamics

Christian Lessig, Ilaria Luise, Martin Schultz, et al.
Large scale representation learning

- Learn a domain-specific but task-independent neural network that is useful for a range of applications
Large scale representation learning

- Learn a domain-specific but task-independent neural network that is useful for a range of applications
  - Representation network provides transformation of network input to effective feature spaces
  - Self-supervised training on very large amounts of data (O (PB)) with very large networks (O (10^{11}) parameters)
  - Useful for downstream applications using tail network, fine-tuning, or in-context learning
Large scale representation learning

- Learn a domain-specific but task-independent neural network that is useful for a range of applications
  - Representation network provides transformation of network input to effective feature spaces
  - Self-supervised training on very large amounts of data (O (PB)) with very large networks (O ($10^{11}$) parameters)
  - Useful for downstream applications using tail network, fine-tuning, or in-context learning
Can we perform representation learning in the Earth sciences?
Representation learning for the Earth sciences?

- Very large amounts of observational data
  - ERA5 reanalysis: 6+ PB
  - ESA’s MetOp-SG satellites: 8 x 864 GB/day
  - Data essentially completely unlabelled
Representation learning for the Earth sciences?

- Very large amounts of observational data
  - ERA5 reanalysis: 6+ PB
  - ESA’s MetOp-SG satellites: 8 x 864 GB/day
  - Data essentially completely unlabelled
Representation learning for the Earth sciences?

- Very large amounts of observational data
  - ERA5 reanalysis: 6+ PB
  - ESA's MetOp-SG satellites: 8 x 864 GB/day
  - Data essentially completely unlabelled

GPT-3: $10^{11}$ tokens
ERA5: $5^{14}$ tokens
Representation learning for the Earth sciences?

- Very large amounts of observational data
- No complete classical model for system and dynamics
  - Central issue for forecasting and climate projections
Representation learning for the Earth sciences?

- Very large amounts of observational data
- No complete classical model for system and dynamics
- Chaoticity in atmospheric dynamics leads to ambiguity
  - There is often not one “correct answer“
  - Large networks learn statistical representations
AtmoRep

Large scale representation learning of atmospheric dynamics
AtmoRep
AtmoRep

Historical observations

- ERA5 reanalysis
- large scale machine learning

1950 1970 1990 2010

Forecasting Impact analysis Downscaling
AtmoRep

Historical observations

ERA5 reanalysis

large scale machine learning

Climate projections

Impact analysis

Forecasting

© Christian Lessig, 2023
AtmoRep

Historical observations

ERA5 reanalysis

large scale machine learning

applications

Forecasting
Impact analysis
Climate projections
Downscaling
AtmoRep

Historical observations

1950 1970 1990 2010

ERA5 reanalysis

large scale machine learning

applications

Forecasting Impact analysis

Climate projections Downscaling

scientific insight

© Christian Lessig, 2023
AtmoRep: a theoretical formulation
AtmoRep: a theoretical formulation

- Atmosphere as abstract stochastical dynamical system:

\[ \bar{p}(\bar{y}|\bar{x}) \]
AtmoRep: a theoretical formulation

- Atmosphere as abstract stochastical dynamical system:

\[ \bar{p}(\bar{y}|\bar{x}) \]
AtmoRep: a theoretical formulation

- Atmosphere as abstract stochastical dynamical system:

\[ p(y|x) \]
AtmoRep: a theoretical formulation

- Atmosphere as abstract stochastical dynamical system:
  \[ \bar{p}(\bar{y} | \bar{x}) \]

- Neural network model:
AtmoRep: a theoretical formulation

- Atmosphere as abstract stochastical dynamical system:

\[ \bar{p}(\bar{y}|\bar{x}) \]

- Neural network model:
AtmoRep: a theoretical formulation

- Atmosphere as abstract stochastical dynamical system:

- Neural network model:

/  
approx. initial condition
AtmoRep: a theoretical formulation

- Atmosphere as abstract stochastical dynamical system:

- Neural network model:

Standard formulation for generative models (e.g., Dall-E, diffusion models)
AtmoRep: a theoretical formulation

- Atmosphere as abstract stochastical dynamical system:

- Neural network model:
  - forecasting
  - downscaling
  - model correction
  - ...

Standard formulation for generative models (e.g., Dall-E, diffusion models)
AtmoRep: a theoretical formulation

- Neural network model as factorization of
AtmoRep: a theoretical formulation

- Neural network model as factorization of
- Intrinsically statistical/probabilistic formulation
  - Fits naturally the statistical/chaotic nature of the atmosphere
AtmoRep: a theoretical formulation

- Neural network model as factorization of
- Intrinsically statistical/probabilistic formulation
  - Fits naturally the statistical/chaotic nature of the atmosphere
- Loss derivation
  - Expectation maximization, ELBO, ...
  - Probabilistic bound on skill of network
AtmoRep: in-context learning

- In-context learning: ability to solve tasks without training with zero-/few-shot evaluation
AtmoRep: in-context learning

- In-context learning: ability to solve tasks without training with zero-/few-shot evaluation
  - Language models: chat programs, translation, auto-correction, ... from training on next sentence prediction task
  - Natural language to specify task
AtmoRep: in-context learning

- In-context learning: ability to solve tasks without training with zero-/few-shot evaluation
  - Language models: chat programs, translation, auto-correction, ... from training on next sentence prediction task
  - Natural language to specify task

What is in-context learning for AtmoRep?
AtmoRep: in-context learning

◦ The model implies that what we want to “control” the output state without learning
AtmoRep: in-context learning

- The model implies that what we want to “control” the output state without learning
AtmoRep: in-context learning

- The model implies that what we want to “control” the output state without learning
  
  - Spatial and temporal location, resolution, quality
  - Few shot: “explain” to the network
AtmoRep

Historical observations

ERA5 reanalysis

large scale machine learning

Forecasting

Impact analysis

Climate projections

Downscaling

scientific insight

© Christian Lessig, 2023
AtmoRep

Historical observations

1950 1970 1990 2010

ERA5 reanalysis

Forecasting

Impact analysis

Climate projections

Downscaling

© Christian Lessig, 2023
AtmoRep network architecture

- Transformer-based as network architecture
  - Scales well to very large data-sets
  - Generative model (with decoder)
  - Attention maps provide (physical) interpretability
AtmoRep network architecture

- Network is local in space-time
AtmoRep network architecture
AtmoRep network architecture
AtmoRep network architecture

- Network is local in space-time
  - Physics of dynamics are universally valid
  - Local particularities can be learned by providing time + space position as auxiliary information
What is a token?
What is a token?
What is a token?
What is a token?

- Token is small neighborhood in space-time
  - Small for token attention / interaction to be informative
  - Big enough so token has rich internal structure
What is a token?

- Token is small neighborhood in space-time
  - Small for token attention / interaction to be informative
  - Big enough so token has rich internal structure
- Token size is field-dependent
Multiformer

Self-Attention MLP Self-Attention MLP Self-Attention MLP
Self-attention

Self-Attention  MLP  Self-Attention  MLP  Self-Attention  MLP

⋯⋯

© Christian Lessig, 2023
Self-attention

vorticity

Self-Attention MLP Self-Attention MLP Self-Attention MLP
Self-attention

divergence

vorticity

Self-Attention  MLP  Self-Attention  MLP  Self-Attention  MLP  ...
Self-Attention

MLP

Self-Attention

MLP

Self-Attention

MLP

Temperature

Divergence

Vorticity

Self-Attention

MLP

Self-Attention

MLP

Self-Attention

MLP

Self attention
Self-attention

Self-Attention
MLP
Self-Attention
MLP
Self-Attention
MLP

geopotential
temperature
divergence
vorticity
Self-attention

gopotential

temperature

divergence

vorticity

Self-Attention
MLP

Self-Attention
MLP

Self-Attention
MLP

Multiformer
Self-Attention

MLP

Self-Attention

MLP

Self-Attention

MLP

Self-Attention

MLP

Self-Attention

MLP

geopotential

temperature

divergence

vorticity

Cross attention

Multiformer
Self-Attention

MLP

geopotential

temperature

divergence

Self-Attention

MLP

vorticity

Self-Attention

MLP

Self-Attention

MLP

Self-Attention

MLP

Self-Attention

MLP

Self-Attention

MLP

Cross attention
Self-Attention
MLP
Self-Attention
MLP
Self-Attention
MLP

geopotential

temperature
divergence

vorticity

Self-Attention
MLP
Self-Attention
MLP
Self-Attention
MLP

Cross attention
Multiformer

◦ Different physical fields with different properties have separate latent spaces (and transformations for these)
◦ Individual fields can be pre-trained independently
◦ Plug-and-play of fields
  › Fields can be added/removed with limited (or no) computational effort
◦ Cross-attention allows for explicit introspection of interaction between fields
Data: ERA5 reanalysis

721x1440 horizontal grid (0.25 degree)

137 vertical layers

over 6 PB of data readily amenable to machine learning

- vorticity
- divergence
- temperature
- geopotential
- ...

hourly for 70 years
Training

- Unbiased hierarchical Monte Carlo sampling of all possible ERA5 space-time cubes
  - Random sampling of (year,month) tuples corresponding to individual files
  - Random sampling of space-time cubes in tuples
  - Trivially parallelizable with one data loader per field
- Area preserving sampling for sphere/Earth to compensate for distortion of regular grid
Spatio-temporal BERT

- Self-supervised training with variation of BERT masked language (or token) model
Spatio-temporal BERT

Flatland view
Spatio-temporal BERT

Flatland view of BERT
Spatio-temporal BERT

- Self-supervised training with variation of BERT masked language language model
  - Natural interpretation as forecasting / hindcasting / interpolation
Spatio-temporal BERT

- Self-supervised training with variation of BERT masked language language model
  - Natural interpretation as forecasting / hindcasting / interpolation
  - Random masking and distortions (noising, coarsening) ensures that a probabilistic model is learned
Statistical loss

- Machine learning: Training on MSE loss is problematic in terms of training dynamics
  - One reason for overly smooth predictions
Statistical loss

- Machine learning: Training on MSE loss is problematic in terms of training dynamics
- Training on just the mean is sub-optimal to learn a probabilistic/statistical representation of the dynamics and the system
Statistical loss
Statistical loss
Statistical loss
Statistical loss
Statistical loss

![Diagram of a tail network with Self-Attention layers and MLPs, leading to a statistical fit.

First order statistical fit]
Statistical loss
Statistical loss
Statistical loss
Statistical loss

end-to-end training encourages statistical representation
Statistical loss: experiments

- BERT with conditional masking
- 975 hPa (high frequency) vorticity
- 40 years of training data
Statistical loss

MSE test loss

epoch

no ensemble, MSE
Statistical loss

MSE test loss

- no ensemble, MSE
- ensemble=10, MSE+stats
Statistical loss

- Predictions:
Statistical loss

- Predictions:
Statistical loss

- Predictions:
Statistical loss

2D Histogram of $L_2$ error vs. std. dev.
Zero shot evaluation

- Evaluate performance on representation network as is
Zero shot evaluation
Zero shot evaluation
Zero shot evaluation
Zero shot performance

![Graph showing zero shot performance with MSE on the y-axis and persistence on the x-axis. The graph indicates a high MSE value.]
Zero shot performance

- MSE in training
- Persistence
Zero shot performance

MSE

- in training
- zero shot, embedding
- persistence
Zero shot performance

![Diagram showing MSE for different training and zero shot conditions.]
Zero shot performance

MSE

BERT

0.35, conditional
0.25, conditional
0.15, conditional
0.15, no ensemble
0.075, conditional
1 token, no conditional
Zero shot performance

![Graph showing MSE for BERT with different configurations]

- 0.35, conditional
- 0.25, conditional
- 0.15, conditional
- 0.15, no ensemble
- 0.075, conditional
- 1 token, no conditional

- Final test loss
Zero shot performance

BERT (0.15) per training data

MSE

- 40 years
- 10 years
- 5 years
- 2 years
Zero shot performance

BERT (0.15) per training data

- Final test loss
- 40 years
- 10 years
- 5 years
- 2 years

MSE
AtmoRep: longer term objectives

○ Weather forecasting
○ Climate projections
○ Coupled Earth system
○ Scientific model
○ Training/fine-tuning on direct observational data
Current/next steps

- Complete representation learning model
  - Scale data and network size
  - Different training tasks and protocols
Current/next steps

- Complete representation learning model
- Downstream applications
  - Weather forecasting
  - Downscaling
  - Model correction
  - ...
AtmoRep

Large scale representation learning of atmospheric dynamics

address climate change

large scale machine learning

scientific insight
AtmoDist: evaluation
AtmoRep data
AtmoRep data

Normalized spectra at 975 hPa

- vorticity
- divergence
- geopotential
- temperature
ERA5 versus ImageNet

![Graph showing the comparison between ERA5 and ImageNet with the y-axis labeled |\xi| and the x-axis labeled with values from 20 to 140. The graph includes two curves: one for ImageNet and another for Vorticity.]
ERA5 versus ImageNet

- stream function
- velocity potential
- velocity
- vector field
- vorticity
- divergence
ERA5 versus ImageNet

\[ |\xi| \]

- ImageNet
- \( \approx \) velocity
ERA5 versus ImageNet

![Graph showing comparison between ERA5 and ImageNet](graph.png)
ERA5 versus ImageNet

- stream function
- velocity potential
- velocity
- vector field
- vorticity
- divergence
Embedding of tokens

multiformer
Embedding of tokens

data loader → embed → multiformer → tail
Embedding of tokens

data loader → embed → multiformer → tail

© Christian Lessig, 2023
Embedding network

- Multiformer models longer range effects and field interactions in a rich latent space
Embedding network

- Multiformer models longer range effects and field interactions in a rich latent space
- Embedding network provides rich encoding of input field
Embedding network

◦ Multiformer models longer range effects and field interactions in a rich latent space
  › Embedding network provides rich encoding of input field
  › Embedding network allows for multi-resolution representation per field, i.e. different token sizes
Embedding of tokens
Embedding of tokens

- Multiformer models longer range effects and field interactions in a rich latent space
  - Embedding network provides rich encoding of input field
  - Embedding network allows for multi-resolution representation per field, i.e. different token sizes
Embedding of tokens

- Multiformer models longer range effects and field interactions in a rich latent space
  - Embedding network provides rich encoding of input field
  - Embedding network allows for multi-resolution representation per field, i.e. different token sizes

Use transformer as embedding network
Embedding of tokens
Embedding of tokens
Embedding of tokens
Embedding of tokens
Embedding of tokens
Embedding of tokens
Training

- Unbiased hierarchical Monte Carlo sampling of all possible ERA5 space-time cubes
Training

1979 $t$ \rightarrow 2020
Training

1979 \[t\] 

03/1984 09/2003 

2020
Training

03/1984  09/2003
Training

1979 \( t \) 03/1984 09/2003 2020
Training

area preserving sampling of sphere

1979 \( t \) \rightarrow 2020

03/1984 09/2003
Training

area preserving sampling of sphere

1979 $t$ 03/1984 09/2003 2020
Statistical loss
Statistical loss

Diagram: A bar chart showing the loss for 'cat' and 'dog'. The bar for 'cat' is higher than the bar for 'dog' on the vertical axis labeled '1'.
Statistical loss
Statistical loss
Statistical loss
Statistical loss

- Statistical loss:
Statistical loss

- Statistical loss:
Statistical loss

\[ \text{ensemble} = 10, \text{MSE} + \text{CRPS} \]

\[ \text{ensemble} = 10, \text{MSE} + \text{stats} \]

MSE test loss vs. epoch
Statistical loss

Histogram of ensemble errors
Zero shot evaluation
Zero shot evaluation
Zero shot evaluation
Zero shot evaluation
Zero shot evaluation
Zero shot evaluation
Zero shot evaluation
AtmoRep data

temperature
AtmoRep data

ggeopotential
AtmoRep data

vorticity
AtmoRep data

divergence
AtmoRep data

![Graph showing data comparison between ImageNet and Vorticity]
Spatio-temporal BERT
Spatio-temporal BERT
Spatio-temporal BERT
Statistical loss

- Attention maps:
3.1 Encoder and Decoder Stacks

Encoder: The encoder is composed of a stack of \( N = 6 \) identical layers. Each layer has two sub-layers. The first is a multi-head self-attention mechanism, and the second is a simple, position-wise fully connected feed-forward network. We employ a residual connection around each of the two sub-layers, followed by layer normalization. That is, the output of each sub-layer is \( \text{LayerNorm}(x + \text{Sublayer}(x)) \), where \( \text{Sublayer}(x) \) is the function implemented by the sub-layer itself. To facilitate these residual connections, all sub-layers in the model, as well as the embedding layers, produce outputs of dimension \( d_{\text{model}} = 512 \).

Decoder: The decoder is also composed of a stack of \( N = 6 \) identical layers. In addition to the two sub-layers in each encoder layer, the decoder inserts a third sub-layer, which performs multi-head attention over the output of the encoder stack. Similar to the encoder, we employ residual connections around each of the sub-layers, followed by layer normalization. We also modify the self-attention sub-layer in the decoder stack to prevent positions from attending to subsequent positions. This masking, combined with the fact that the output embeddings are offset by one position, ensures that the predictions for position \( i \) can depend only on the known outputs at positions less than \( i \).

3.2 Attention

An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key.

Forecasting / projections

3.1 Encoder and Decoder Stacks

Encoder: The encoder is composed of a stack of $N = 6$ identical layers. Each layer has two sub-layers. The first is a multi-head self-attention mechanism, and the second is a simple, position-wise fully connected feed-forward network. We employ a residual connection \([ \text{LayerNorm}(x + \text{Sublayer}(x)) ]\) around each of the two sub-layers, followed by layer normalization \([ \text{LayerNorm}(x) ]\). That is, the output of each sub-layer is \(\text{LayerNorm}(x + \text{Sublayer}(x))\), where \(\text{Sublayer}(x)\) is the function implemented by the sub-layer itself. To facilitate these residual connections, all sub-layers in the model, as well as the embedding layers, produce outputs of dimension \(d_{\text{model}} = 512\).

Decoder: The decoder is also composed of a stack of $N = 6$ identical layers. In addition to the two sub-layers in each encoder layer, the decoder inserts a third sub-layer, which performs multi-head attention over the output of the encoder stack. Similar to the encoder, we employ residual connections around each of the sub-layers, followed by layer normalization. We also modify the self-attention sub-layer in the decoder stack to prevent positions from attending to subsequent positions. This masking, combined with the fact that the output embeddings are offset by one position, ensures that the predictions for position $i$ can depend only on the known outputs at positions less than $i$.

3.2 Attention

An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key.

3.1 Encoder and Decoder Stacks

Encoder:
The encoder is composed of a stack of $N = 6$ identical layers. Each layer has two sub-layers. The first is a multi-head self-attention mechanism, and the second is a simple, position-wise fully connected feed-forward network. We employ a residual connection around each of the two sub-layers, followed by layer normalization. That is, the output of each sub-layer is $\text{LayerNorm}(x + \text{Sublayer}(x))$, where $\text{Sublayer}(x)$ is the function implemented by the sub-layer itself. To facilitate these residual connections, all sub-layers in the model, as well as the embedding layers, produce outputs of dimension $d_{\text{model}} = 512$.

Decoder:
The decoder is also composed of a stack of $N = 6$ identical layers. In addition to the two sub-layers in each encoder layer, the decoder inserts a third sub-layer, which performs multi-head attention over the output of the encoder stack. Similar to the encoder, we employ residual connections around each of the sub-layers, followed by layer normalization. We also modify the self-attention sub-layer in the decoder stack to prevent positions from attending to subsequent positions. This masking, combined with the fact that the output embeddings are offset by one position, ensures that the predictions for position $i$ can depend only on the known outputs at positions less than $i$.

3.2 Attention

An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key.


Forecasting / projections

autoregressive, generative modeling

akin to time stepping loop (roll out) for forecasting/projections
Forecasting / projections

3.1 Encoder and Decoder Stacks

Encoder:
The encoder is composed of a stack of $N = 6$ identical layers. Each layer has two sub-layers. The first is a multi-head self-attention mechanism, and the second is a simple, position-wise fully connected feed-forward network. We employ a residual connection around each of the two sub-layers, followed by layer normalization. That is, the output of each sub-layer is $\text{LayerNorm}(x + \text{Sublayer}(x))$, where $\text{Sublayer}(x)$ is the function implemented by the sub-layer itself. To facilitate these residual connections, all sub-layers in the model, as well as the embedding layers, produce outputs of dimension $d_{\text{model}} = 512$.

Decoder:
The decoder is also composed of a stack of $N = 6$ identical layers. In addition to the two sub-layers in each encoder layer, the decoder inserts a third sub-layer, which performs multi-head attention over the output of the encoder stack. Similar to the encoder, we employ residual connections around each of the sub-layers, followed by layer normalization. We also modify the self-attention sub-layer in the decoder stack to prevent positions from attending to subsequent positions. This masking, combined with fact that the output embeddings are offset by one position, ensures that the predictions for position $i$ can depend only on the known outputs at positions less than $i$.

3.2 Attention

An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key.

3.1 Encoder and Decoder Stacks

**Encoder:** The encoder is composed of a stack of $N = 6$ identical layers. Each layer has two sub-layers. The first is a multi-head self-attention mechanism, and the second is a simple, position-wise fully connected feed-forward network. We employ a residual connection \[ \text{LayerNorm}(x + \text{Sublayer}(x)) \] around each of the two sub-layers, followed by layer normalization \[ \text{LayerNorm}(x) \]. That is, the output of each sub-layer is \[ \text{LayerNorm}(x + \text{Sublayer}(x)) \], where \[ \text{Sublayer}(x) \] is the function implemented by the sub-layer itself. To facilitate these residual connections, all sub-layers in the model, as well as the embedding layers, produce outputs of dimension \[ d_{\text{model}} = 512 \].

**Decoder:** The decoder is also composed of a stack of $N = 6$ identical layers. In addition to the two sub-layers in each encoder layer, the decoder inserts a third sub-layer, which performs multi-head attention over the output of the encoder stack. Similar to the encoder, we employ residual connections around each of the sub-layers, followed by layer normalization. We also modify the self-attention sub-layer in the decoder stack to prevent positions from attending to subsequent positions. This masking, combined with the fact that the output embeddings are offset by one position, ensures that the predictions for position $i$ can depend only on the known outputs at positions less than $i$.

3.2 Attention

An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key.

---

**Forecasting / projections**

coarse scale/simple classical model

slow climate variables


autoregressive, generative modeling

akin to time stepping loop (roll out) for forecasting/projections