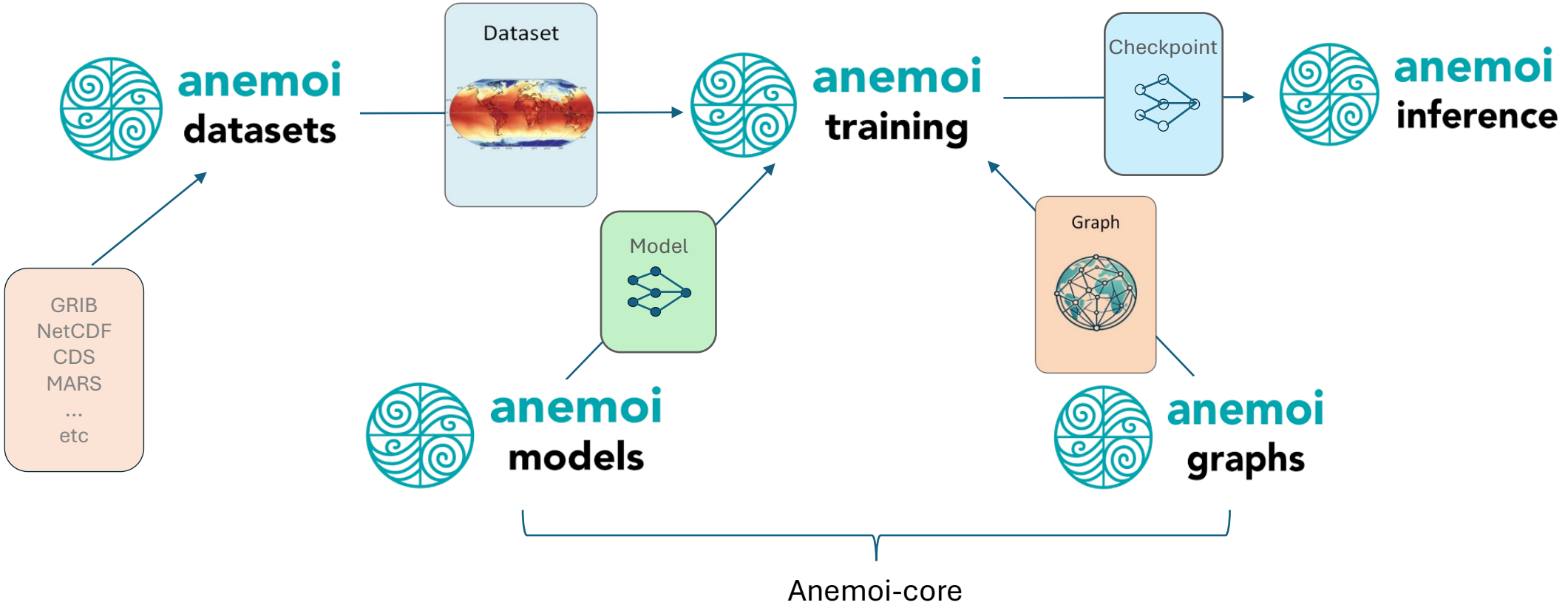
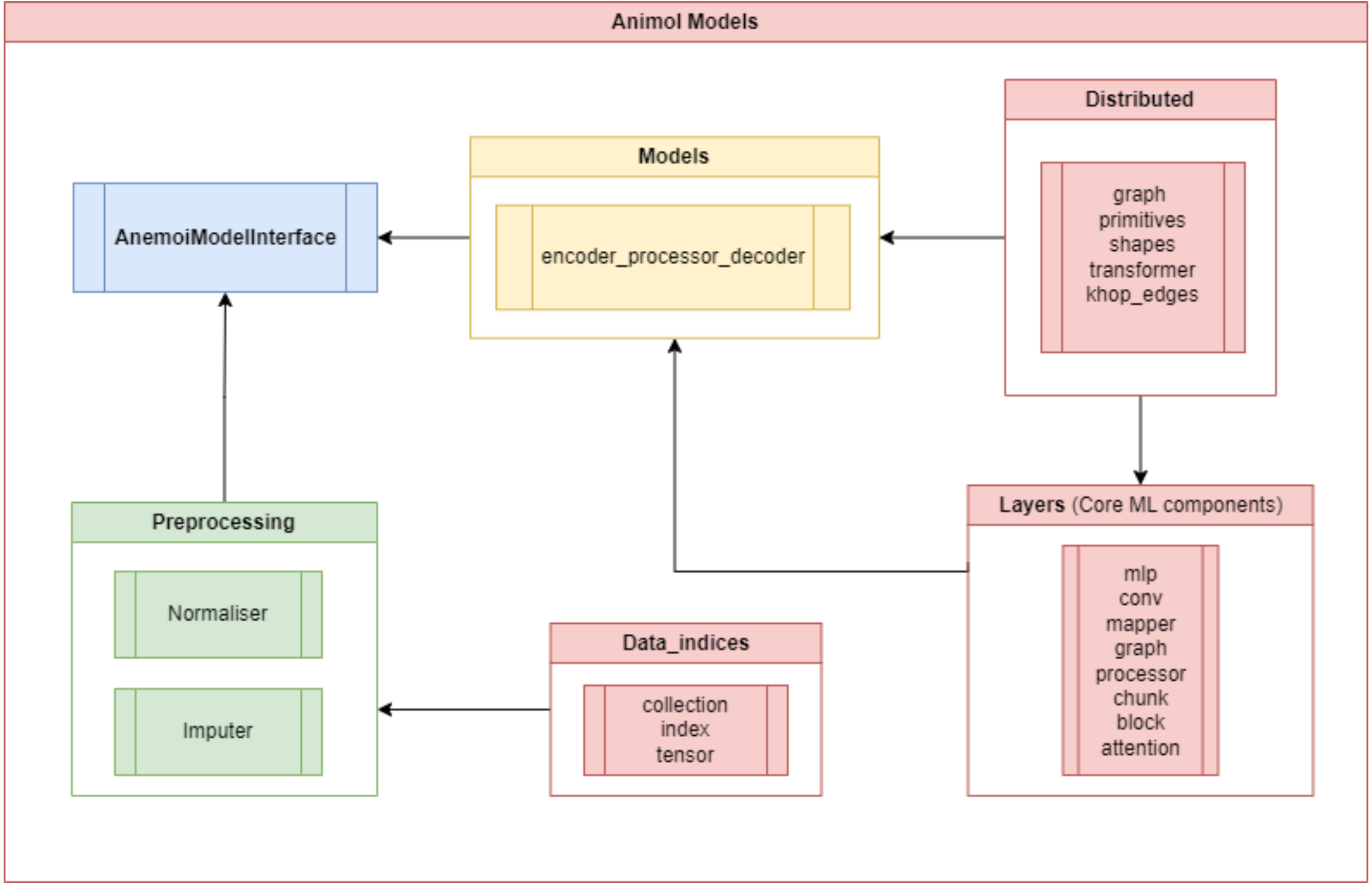
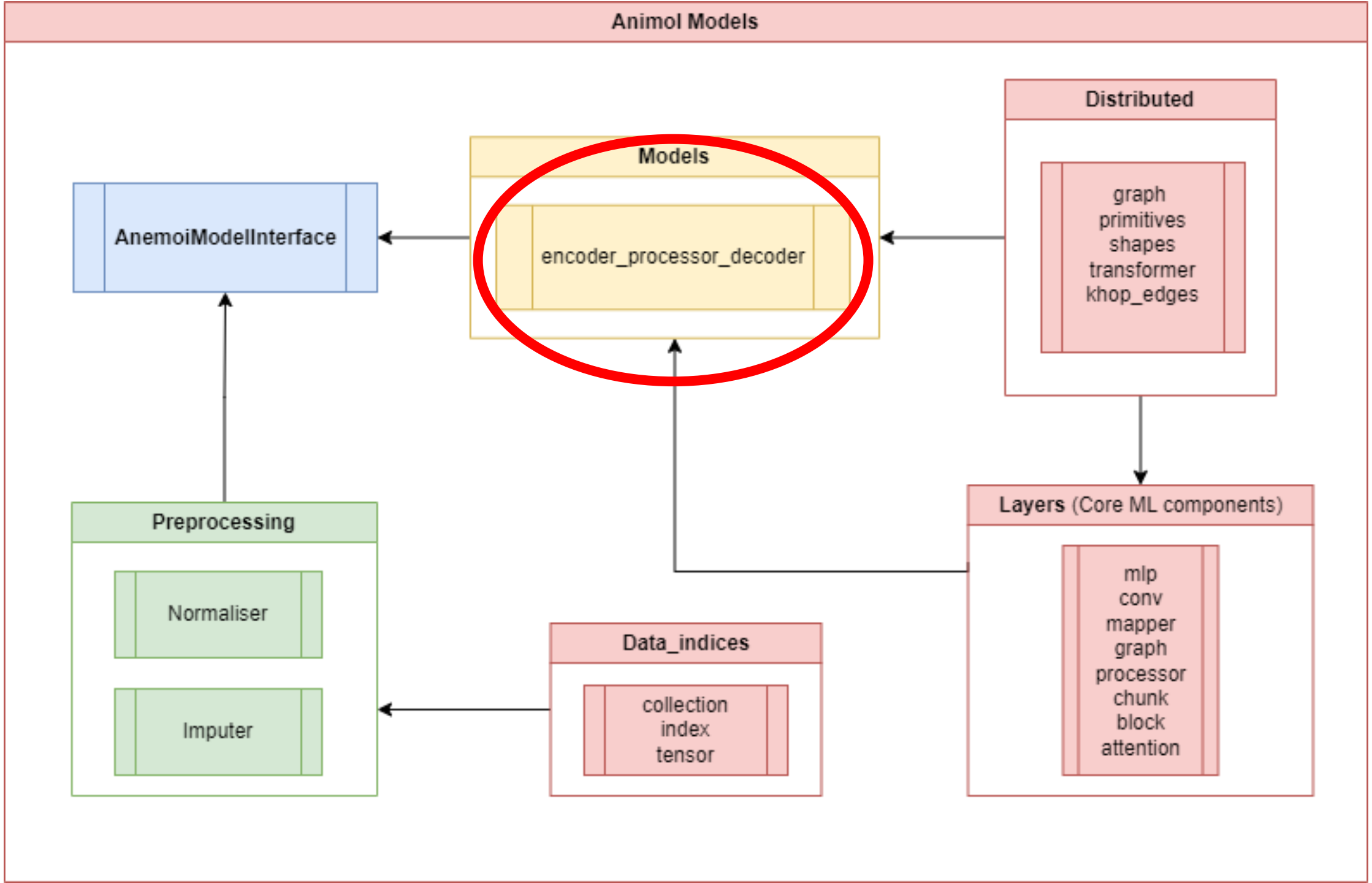
Anemoi Models

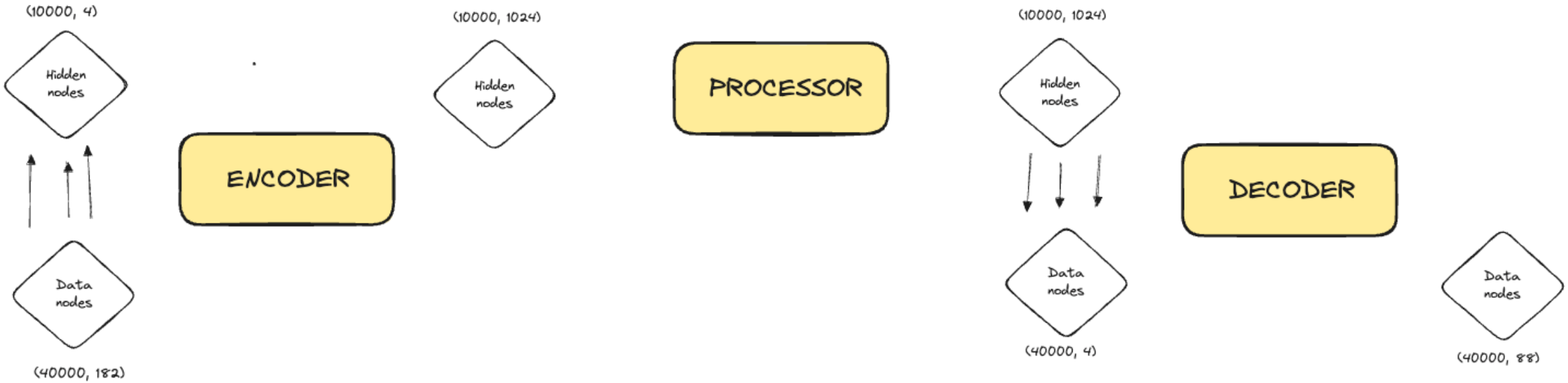
Anemoi in a Nutshell

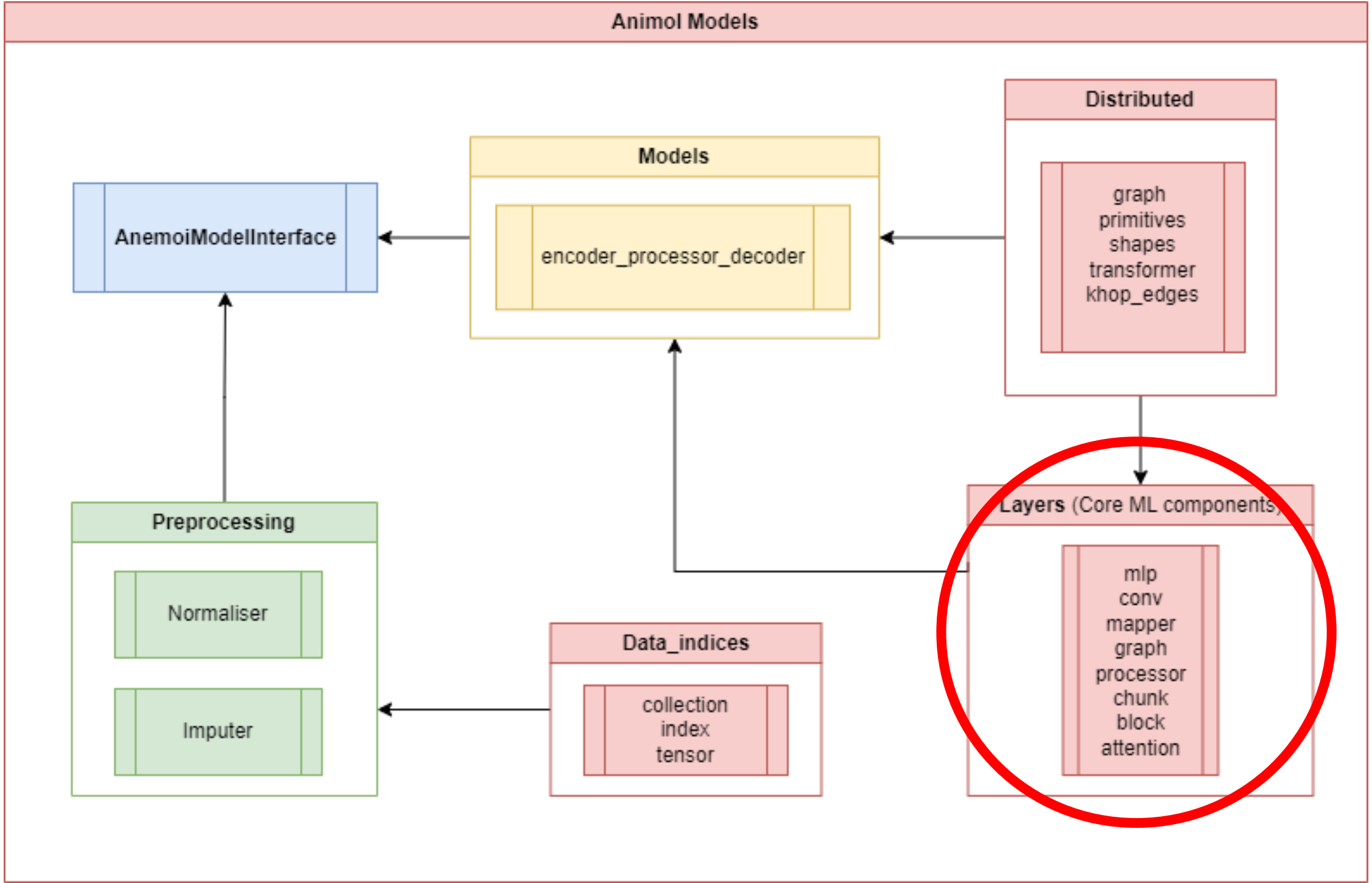
- Anemoi is an open-source framework to develop **data-driven meteorological forecasting**.
- Anemoi is developed through a **collaborative European initiative**




Animol Models
Distributed
graph
primitives
shapes
transformer
khop_edges
Models
encoder_processor_decoder
AnemoiModelInterface
Layers (Core ML components)
mlp
conv
mapper
graph
processor
chunk
block
attention
Preprocessing
Normaliser
Imputer
Data_indices
collection
index
tensor

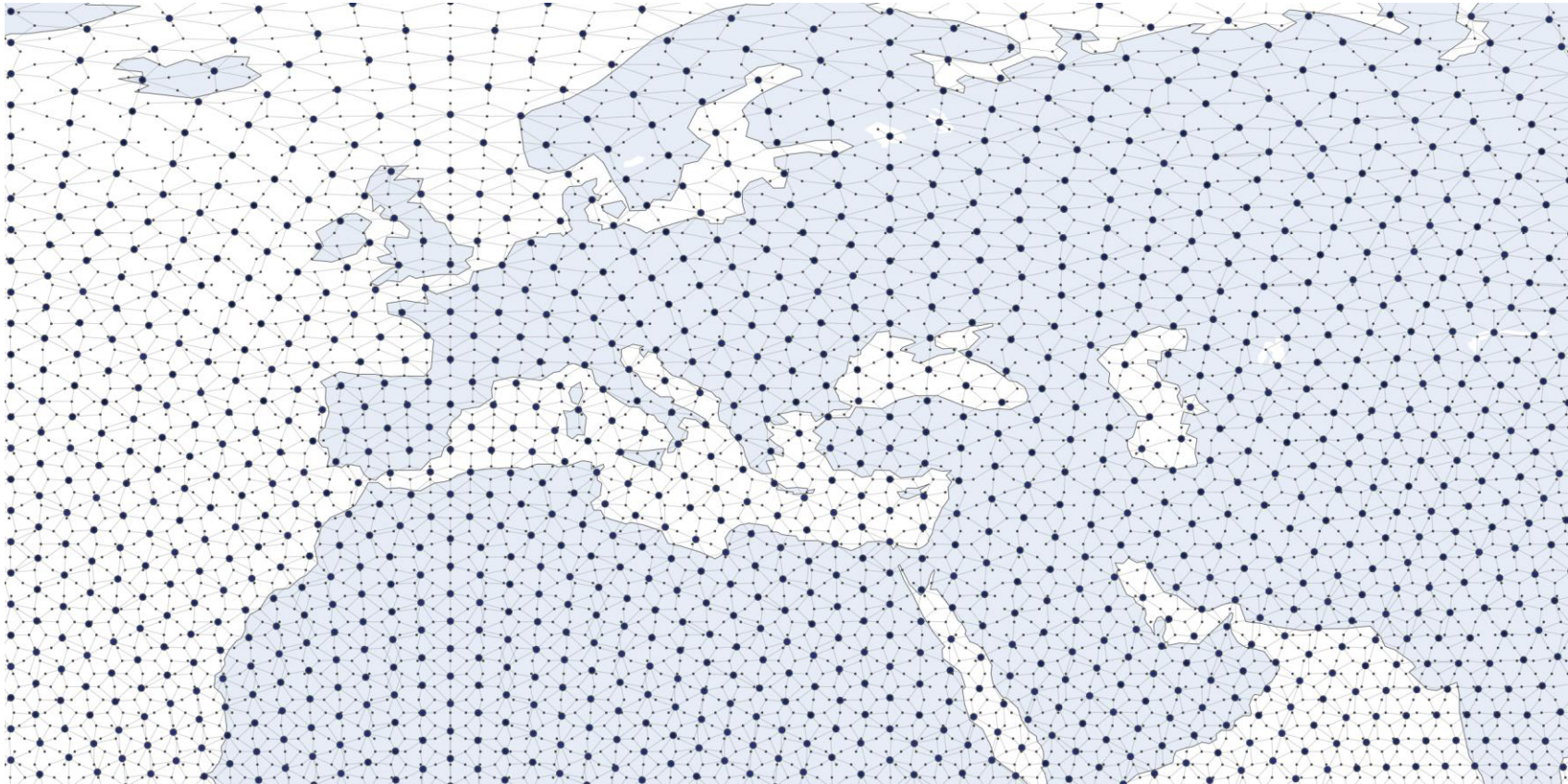

Animol Models
Distributed
graph
primitives
shapes
transformer
khop_edges
Models
encoder_processor_decoder
AnemoiModelInterface
Layers (Core ML components)
mlp
conv
mapper
graph
processor
chunk
block
attention
Preprocessing
Normaliser
Imputer
Data_indices
collection
index
tensor


(10000, 4)
Hidden nodes
Data nodes
(40000, 182)
ENCODER
(10000, 1024)
Hidden nodes
PROCESSOR
(10000, 1024)
Hidden nodes
Data nodes
(40000, 4)
DECODER
Data nodes
(40000, 88)


Animol Models
Distributed
graph
primitives
shapes
transformer
khop_edges
Models
encoder_processor_decoder
AnemoiModelInterface
Layers (Core ML components)
mlp
conv
mapper
graph
processor
chunk
block
attention
Preprocessing
Normaliser
Imputer
Data_indices
collection
index
tensor

How to build an encoder processor decoder?

- Encoders/decoders use Mappers.
  - Map from one grid to another.
  - 3 to choose from:
    - GNN (message passing)
    - **GraphTransformer**
    - Transformer

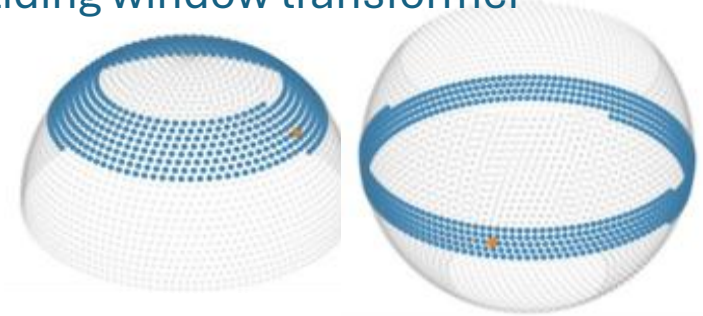


ECMWF

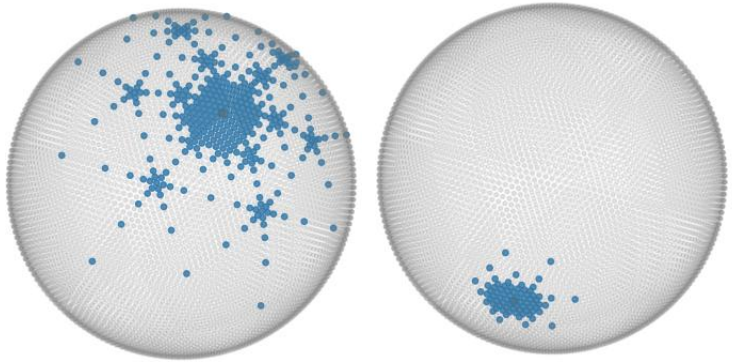
How to build an encoder processor decoder?

- Processors layers to build the processor.
  - Operate on fixed grid.
  - 3 to choose from:
    - GNN (message passing)
    - **GraphTransformer**
    - **Transformer**

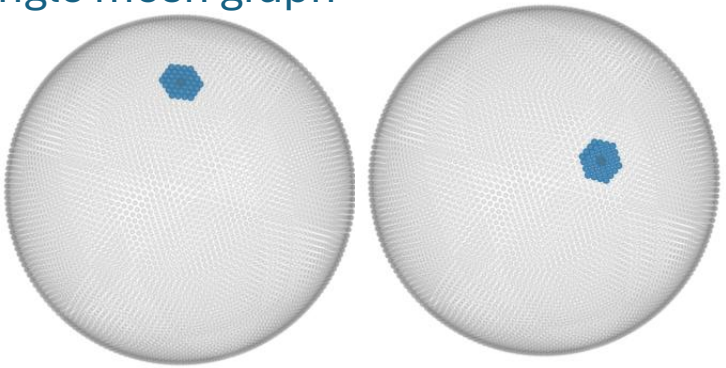
Sliding window transformer



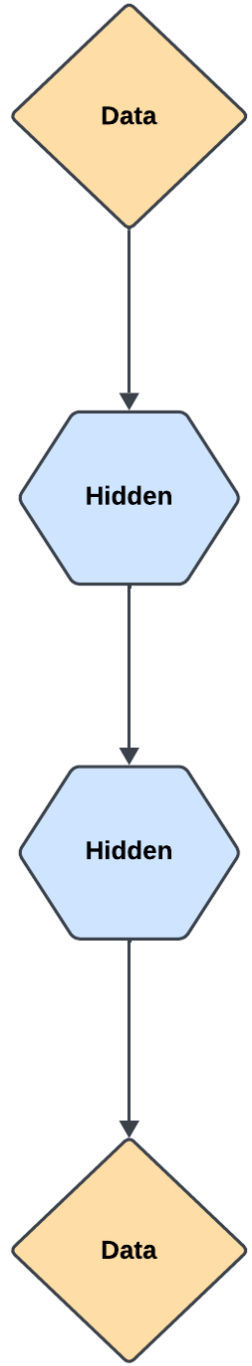
Multi mesh graph

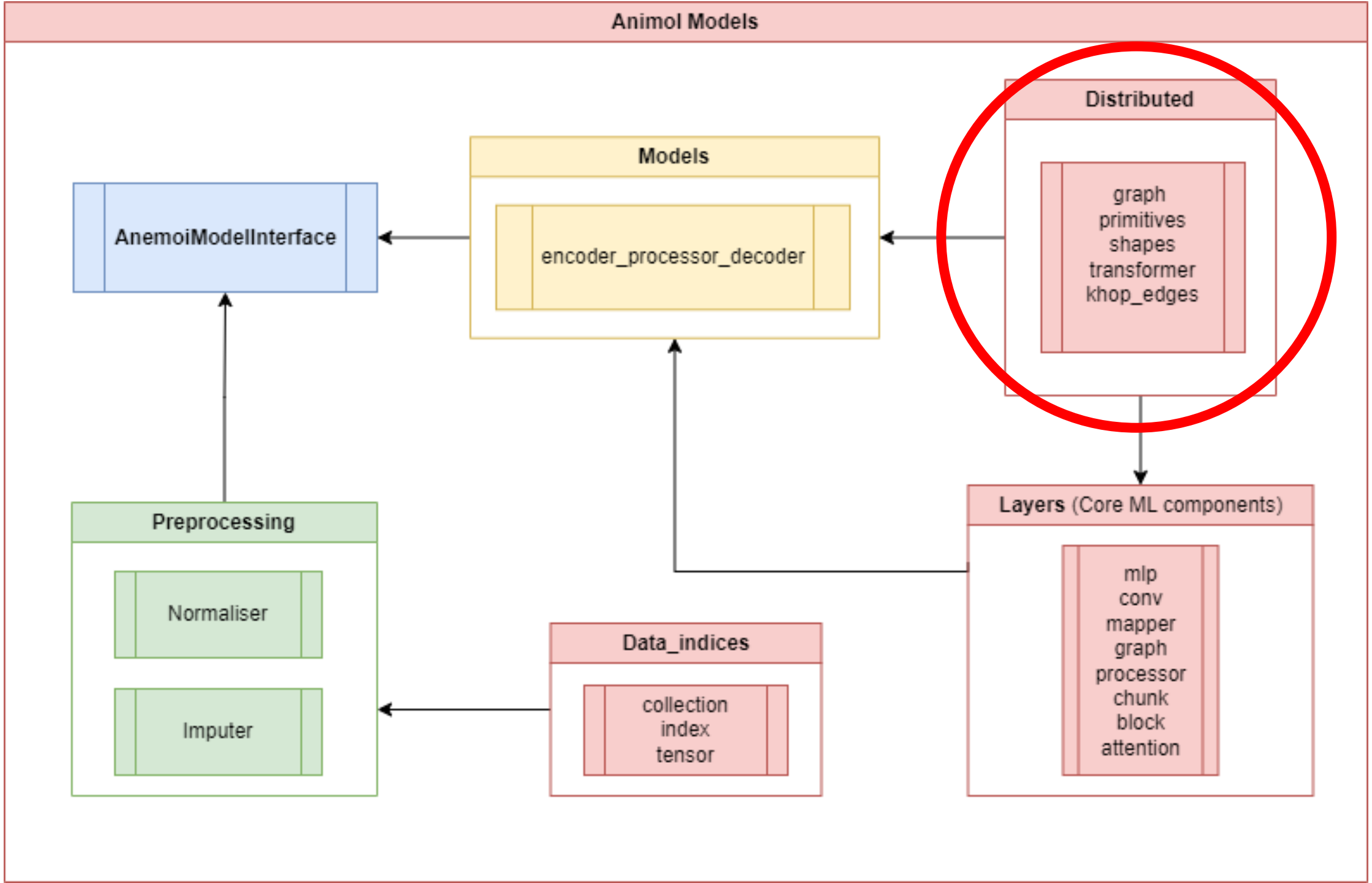


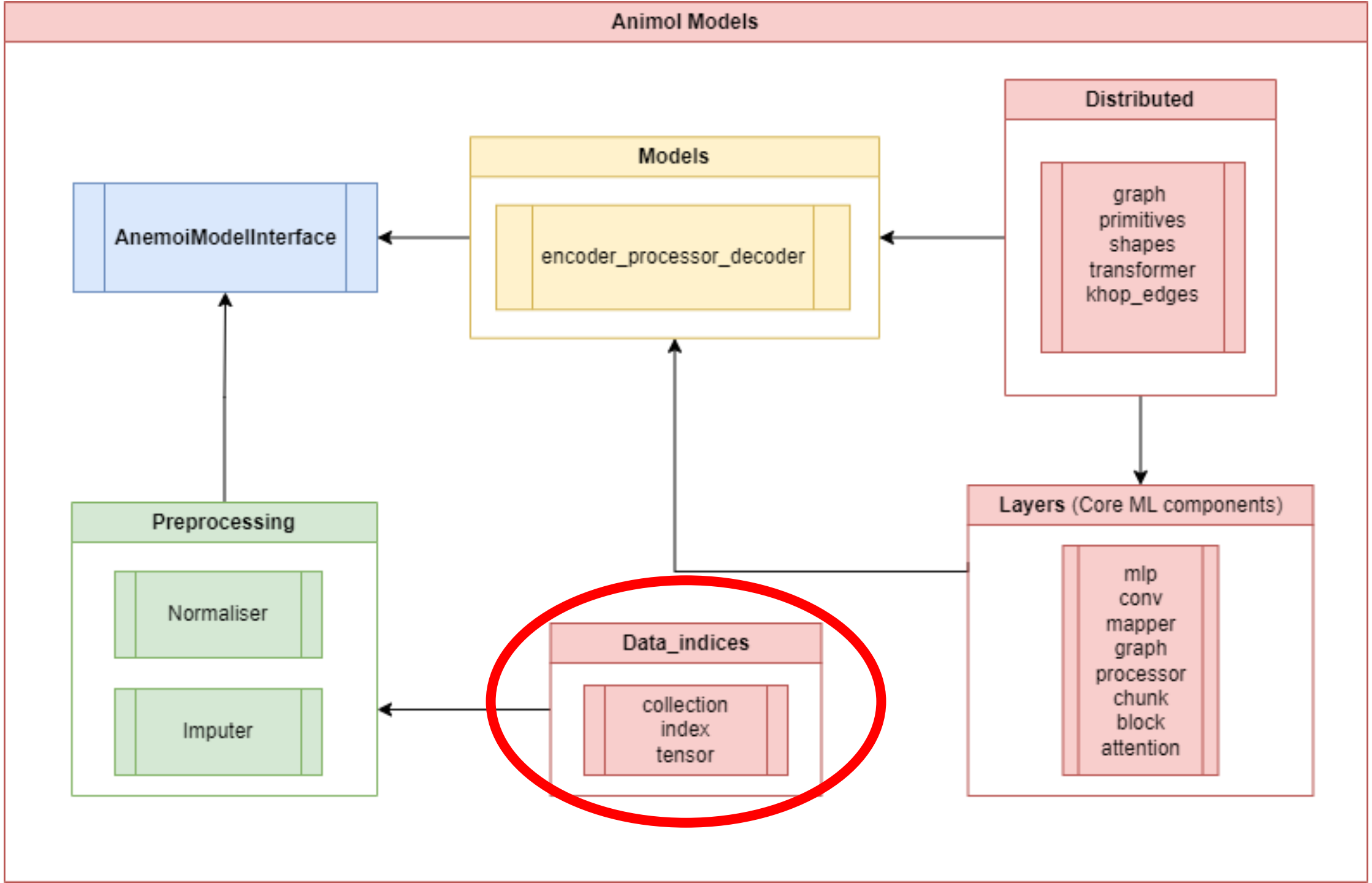
Single mesh graph



```yaml
encoder:
  _target_: anemoi.models.layers.mapper.GraphTransformerForwardMapper
  trainable_size: ${model.trainable_parameters.data2hidden}
  sub_graph_edge_attributes: ${model.attributes.edges}
  num_chunks: 4
  mlp_hidden_ratio: 4 # GraphTransformer or Transformer only
  num_heads: 16 # GraphTransformer or Transformer only
  qk_norm: False
  cpu_offload: ${model.cpu_offload}
  layer_kernels: ${model.layer_kernels}
  shard_strategy: "edges"
processor:
  _target_: anemoi.models.layers.processor.GraphTransformerProcessor
  trainable_size: ${model.trainable_parameters.hidden2hidden}
  sub_graph_edge_attributes: ${model.attributes.edges}
  num_layers: 16
  num_chunks: 4
  mlp_hidden_ratio: 4 # GraphTransformer or Transformer only
  num_heads: 16 # GraphTransformer or Transformer only
  qk_norm: False
  cpu_offload: ${model.cpu_offload}
  layer_kernels: ${model.layer_kernels}
decoder:
  _target_: anemoi.models.layers.mapper.GraphTransformerBackwardMapper
  trainable_size: ${model.trainable_parameters.hidden2data}
  sub_graph_edge_attributes: ${model.attributes.edges}
  num_chunks: 4
  mlp_hidden_ratio: 4 # GraphTransformer or Transformer only
  num_heads: 16 # GraphTransformer or Transformer only
  initialise_data_extractor_zero: False
  qk_norm: False
  cpu_offload: ${model.cpu_offload}
  layer_kernels: ${model.layer_kernels}
  shard_strategy: "edges"
```

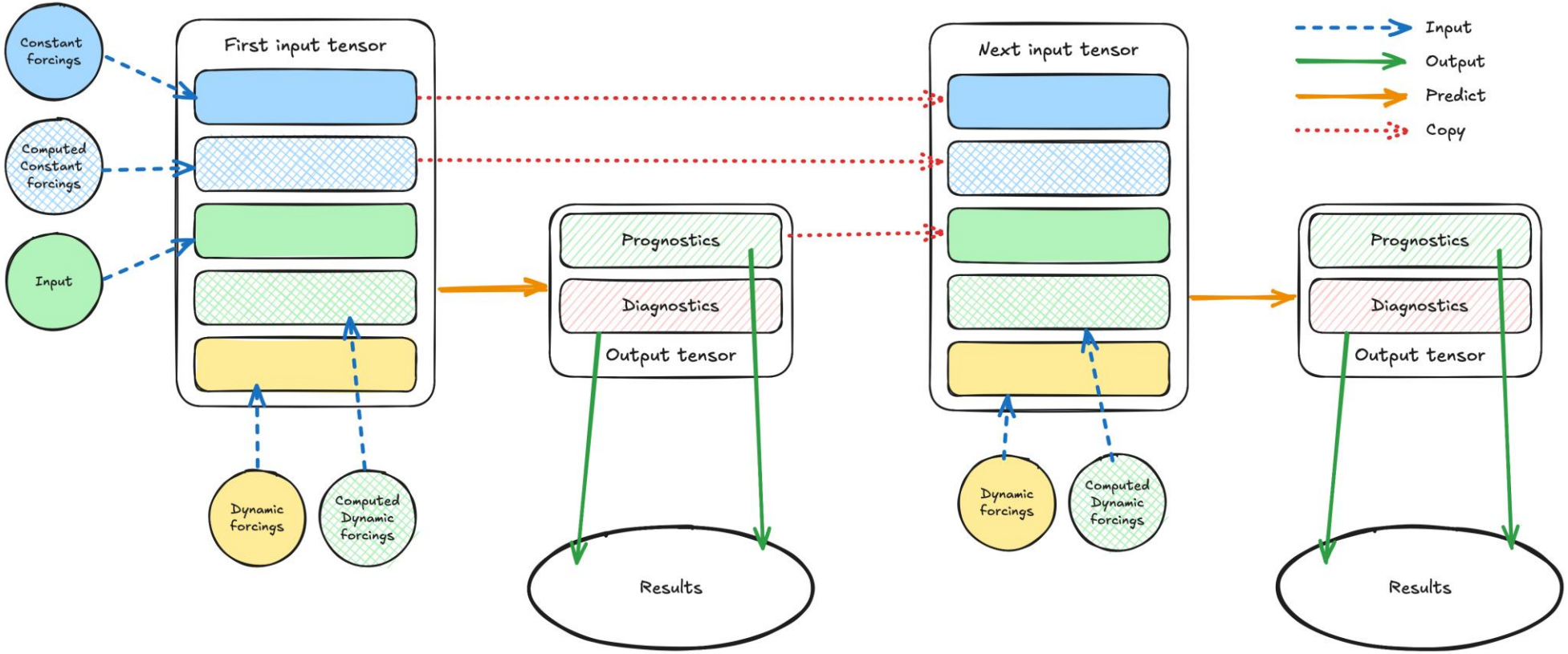



Animol Models
Distributed
graph
primitives
shapes
transformer
khop_edges
Models
encoder_processor_decoder
AnemoiModelInterface
Preprocessing
Normaliser
Imputer
Data_indices
collection
index
tensor
Layers (Core ML components)
mlp
conv
mapper
graph
processor
chunk
block
attention

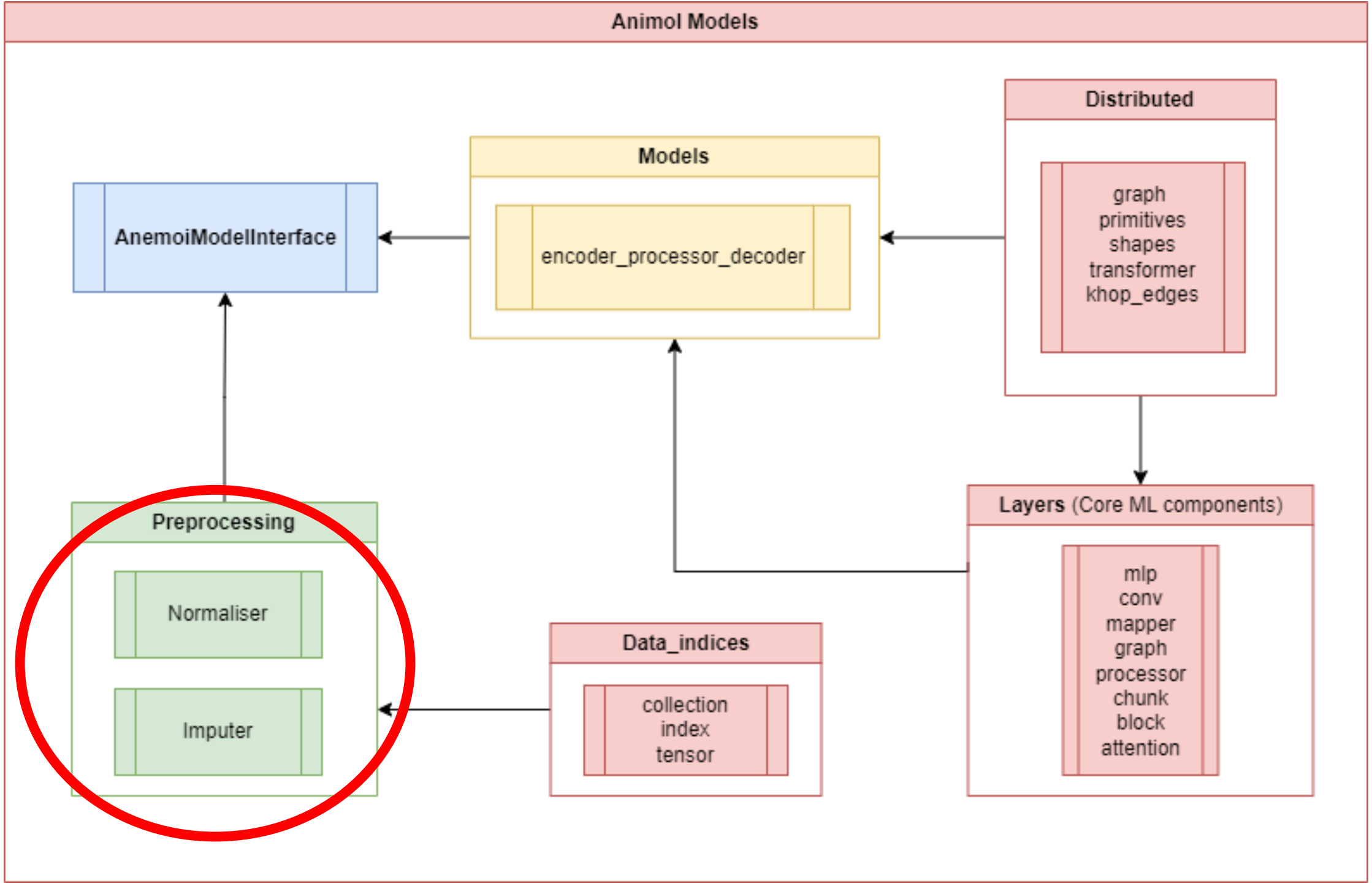

Animol Models
Distributed
graph
primitives
shapes
transformer
khop_edges
Models
encoder_processor_decoder
AnemoiModelInterface
Preprocessing
Normaliser
Imputer
Data_indices
collection
index
tensor
Layers (Core ML components)
mlp
conv
mapper
graph
processor
chunk
block
attention

How to index the big array?

- Anemoi datasets, we built a big array.
- But we may use data differently.
- **Prognostic** (in and out), **forcing** (in), and **diagnostic** (out).
- Need code to understand how to get data in and out of the tensor.



ECMWF


Animol Models
Distributed
graph
primitives
shapes
transformer
khop_edges
Models
encoder_processor_decoder
AnemoiModelInterface
Layers (Core ML components)
mlp
conv
mapper
graph
processor
chunk
block
attention
Preprocessing
Normaliser
Imputer
Data_indices
collection
index
tensor

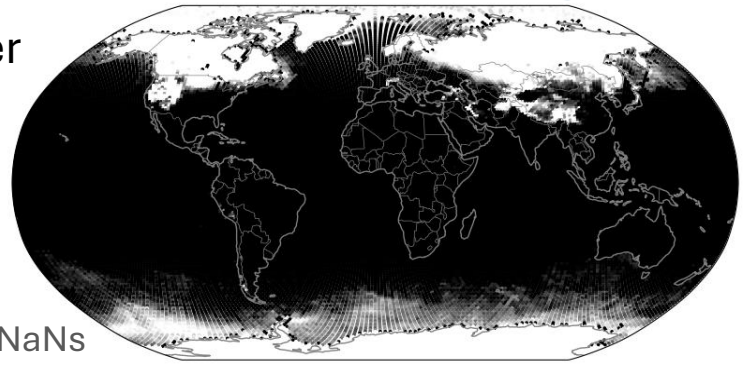
Normaliser

- Apply the statistics to normalise the data.
  - Could be in the training code.
  - But it living in the model helps with inference.
- Options:
  - Mean-std
  - Std
  - Max
  - Min-max
  - None

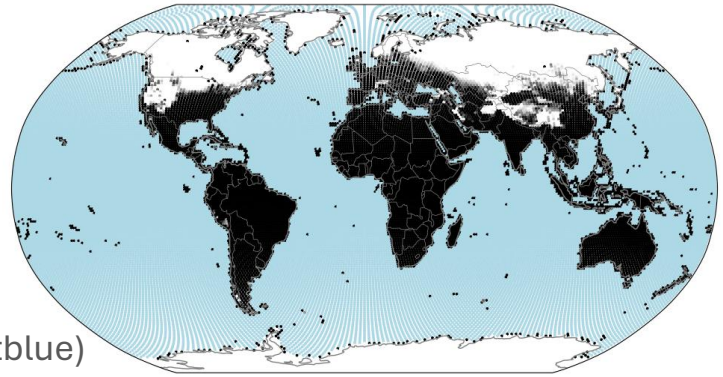
```yaml
normalizer:
  default: "mean-std"

  # Remap cp statistics to those of tp when using FractionBounding. This ensures
  # that cp, as a fraction of tp, remains consistent with tp's scale and statistics.
  # NOTE: This remap should only be applied if FractionBounding is enabled for cp.
  # remap:
  #   cp: tp

  # Standardization applied to tp and cp variables. Ensure that if cp is bounded
  # as a fraction of tp, both variables are normalized using these shared statistics.
  # "Std" normalization is preferred here over "mean-std" to avoid shifting of the
  # zero value in the normalized space.
  std:
  - "tp"
  # - "cp"

  min-max:
  max:
  - "sdor"
  - "slor"
  - "z"
  none:
  - "cos_latitude"
  - "cos_longitude"
  - "sin_latitude"
  - "sin_longitude"
  - "cos_julian_day"
  - "cos_local_time"
  - "sin_julian_day"
  - "sin_local_time"
  - "insolation"
  - "lsm"
```

Dealing with missing values

Snow Cover over ocean



Without masking NaNs



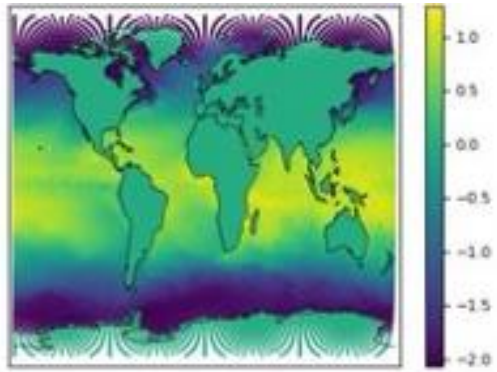
Masking NaNs (lightblue)

In training: NaNs are replaced by values and **masked in the loss function with zeros**

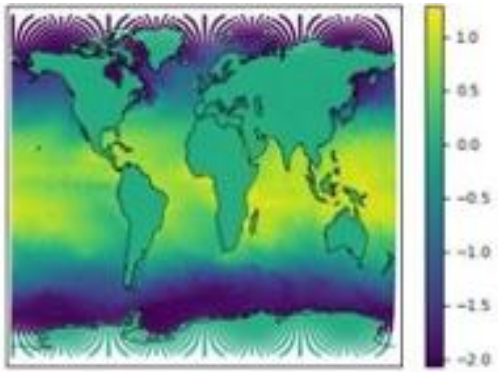
At inference: the model **'predicts'** values **in the masked areas** – these are **removed** in a **post-processing** step

Sea Surface Temperature over land

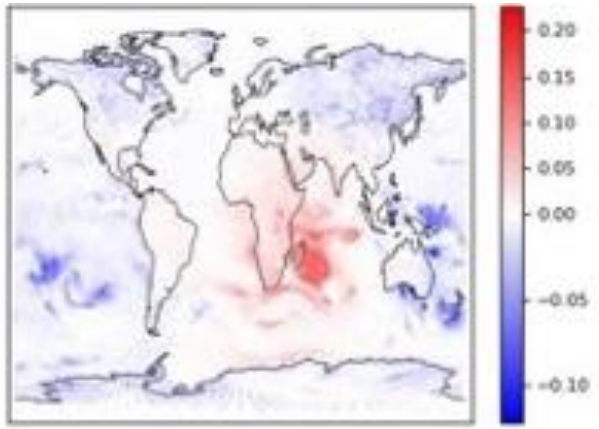
Target



Predicted field

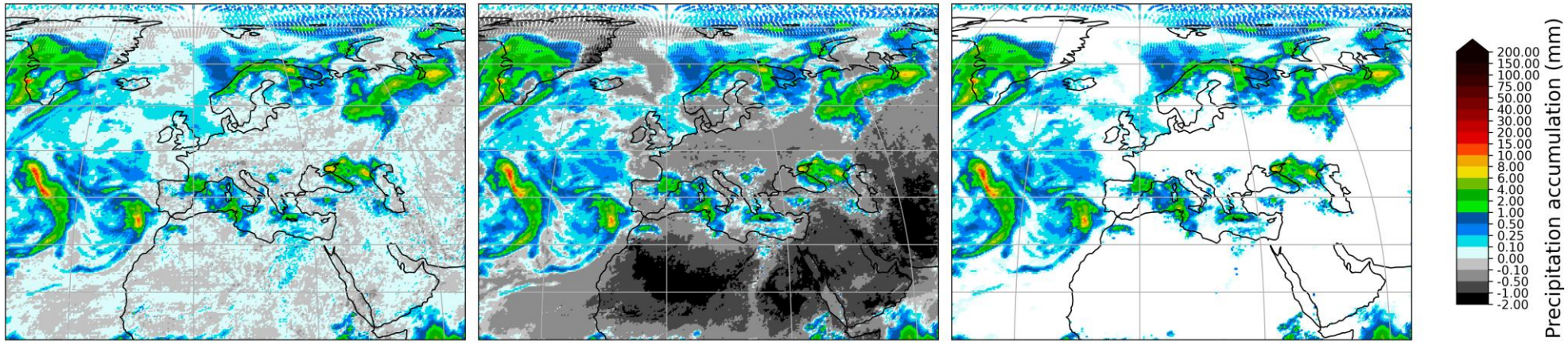


Predicted Increment



- *Additional issues caused when NaNs move! This was the case with waves fields.*

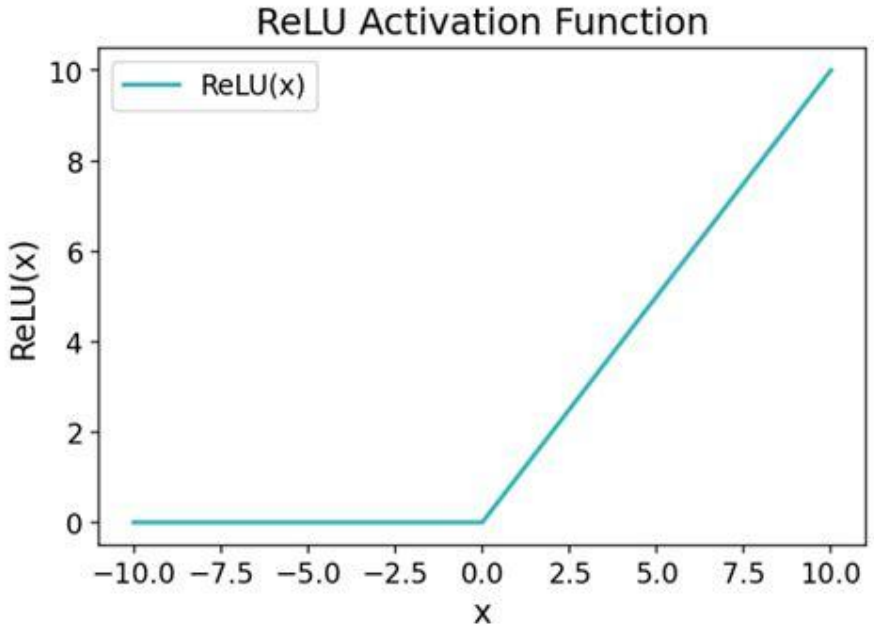
Bounding – a missing piece of the model diagram!



Previous AIFS

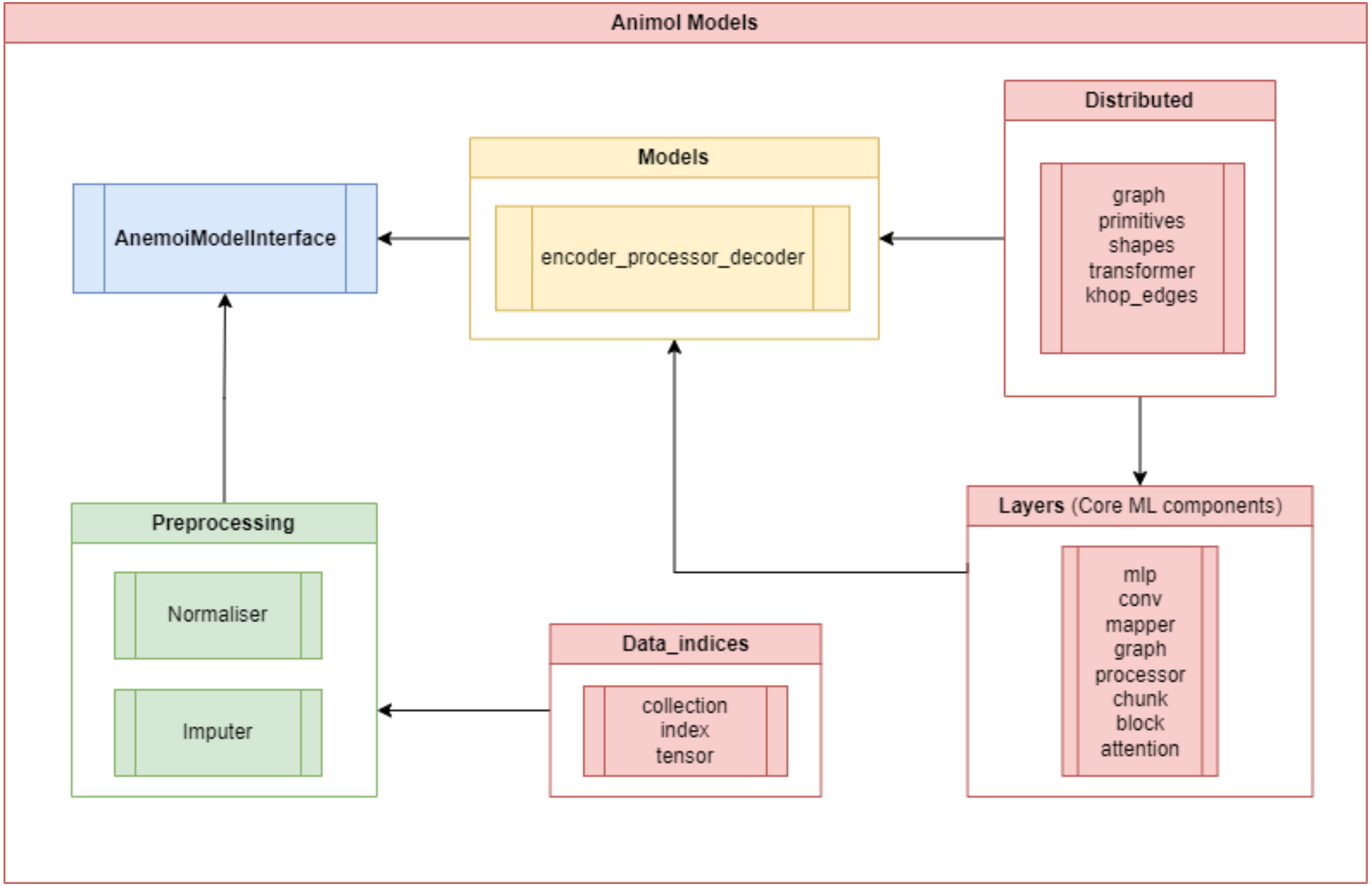
Input to bounding layer

New AIFS




```yaml
bounding: #These are applied in order

# Bound tp (total precipitation) with a Relu bounding layer
# ensuring a range of [0, infinity) to avoid negative precipitation values.
- _target_: anemoi.models.layers.bounding.ReluBounding #[0, infinity)
  variables:
  - tp

# [OPTIONAL] Bound cp (convective precipitation) as a fraction of tp.
# This guarantees that cp is physically consistent with tp by restricting cp
# to a fraction of tp [0 to 1]. Uncomment the lines below to apply.
# NOTE: If this bounding strategy is used, the normalization of cp must be
# changed to "std" normalization, and the "cp" statistics should be remapped
# to those of tp to ensure consistency.

# - _target_: anemoi.models.layers.bounding.FractionBounding # fraction of tp
#   variables:
#   - cp
#   min_val: 0
#   max_val: 1
#   total_var: tp
```

Questions?



ECMWF