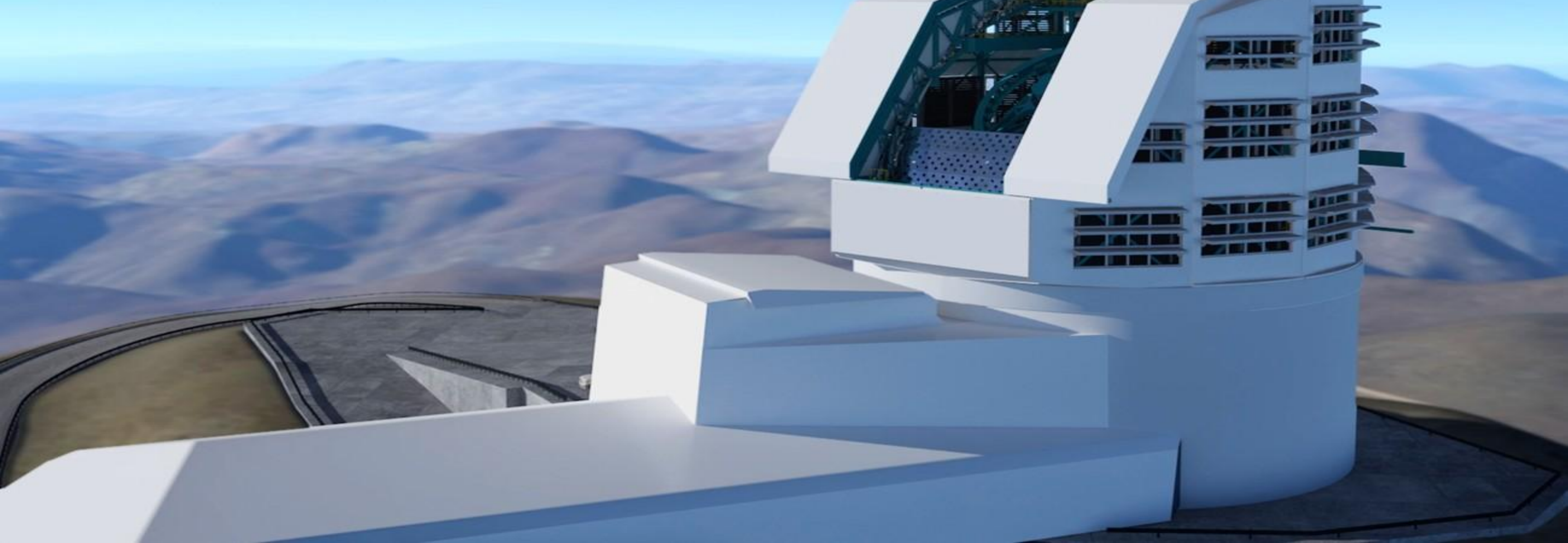
Time-Series Transformers in FINK

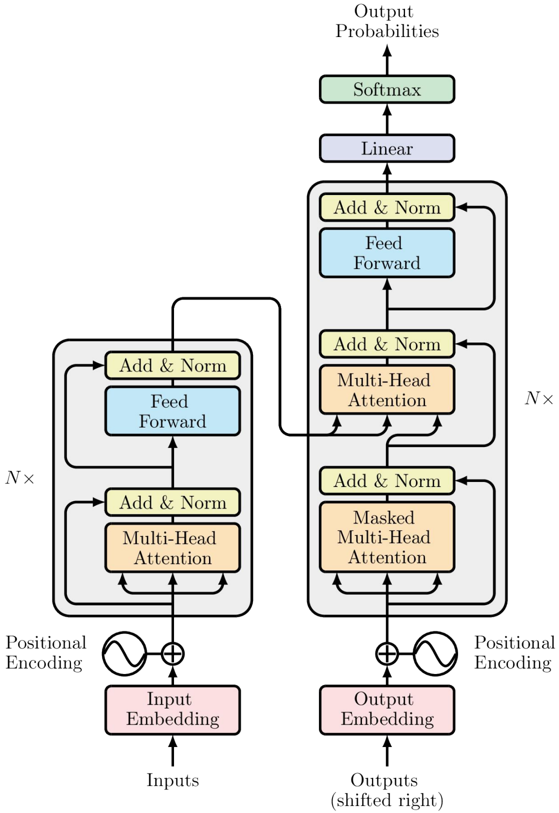
Tarek Allam Jr.
University College London (UCL)



Attention Is All You Need (Vaswani et al. 2017)

Overview

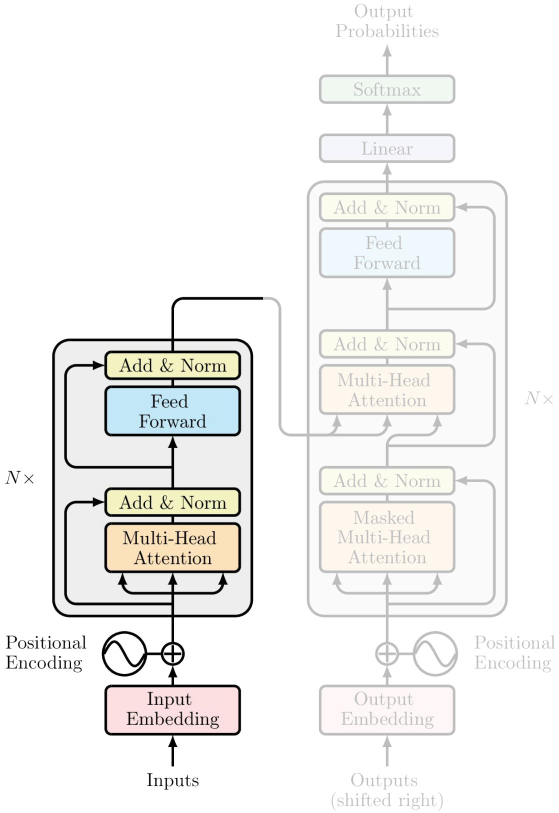
- Breakthrough work in sequence modelling
- Still SOTA for many sequence modelling tasks
- O(1) processing of input, regardless of length, compared to O(n) for RNNs and O(nlogn) for TCNs
- Embarrassingly parallelizable operations



Attention Is All You Need (Vaswani et al. 2017)

Overview

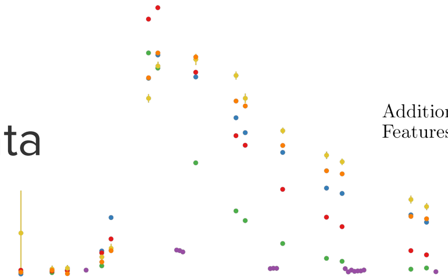
- Breakthrough work in sequence modelling
- Still SOTA for many sequence modelling tasks
- O(1) processing of input, regardless of length, compared to O(n) for RNNs and O(nlogn) for TCNs
- Embarrassingly parallelizable operations

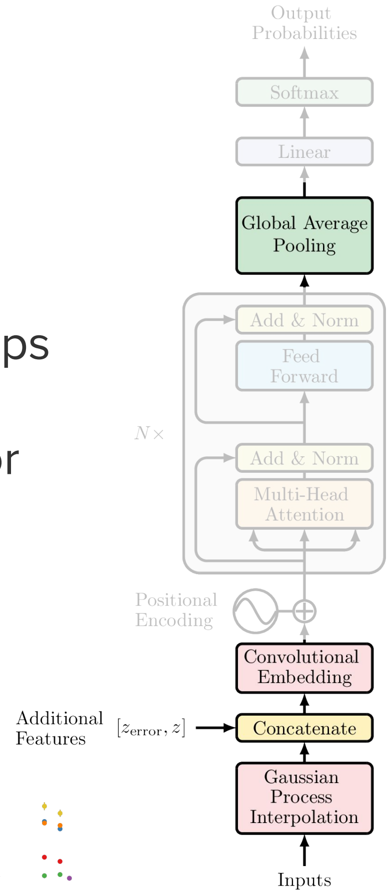


The Time-Series Transformer [t2]

Encoder++

- *Global Average Pooling* to allow for Class Activation Maps
- *Convolutional Embedding* maps time-series into a vector space
- *Concatenate* additional features
- *GP Interpolation* to handle irregular data

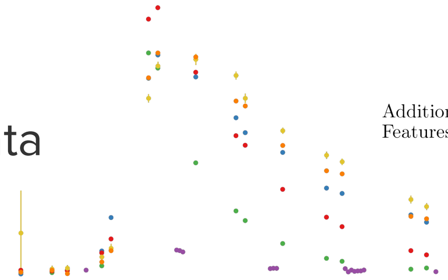


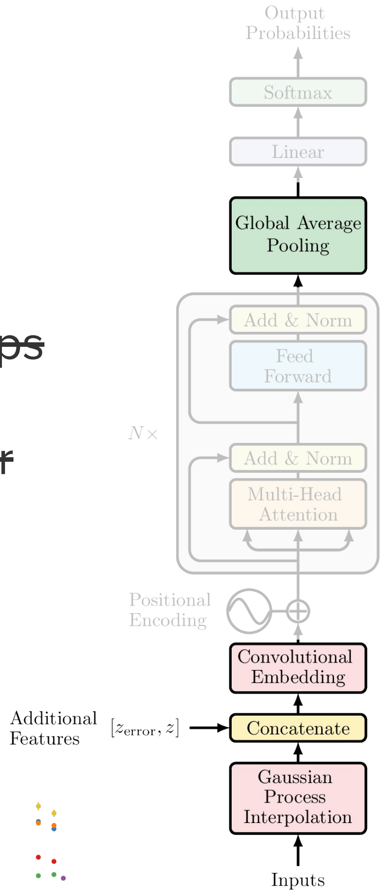


The Time-Series Transformer [t2]

Encoder++

- ~~*Global Average Pooling* to allow for Class Activation Maps~~
- ~~*Convolutional Embedding* maps time-series into a vector space~~
- ~~*Concatenate* additional features~~
- *GP Interpolation* to handle irregular data

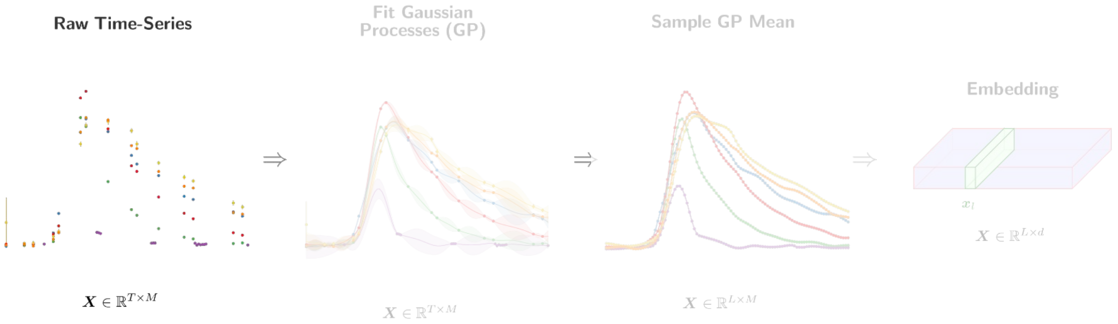


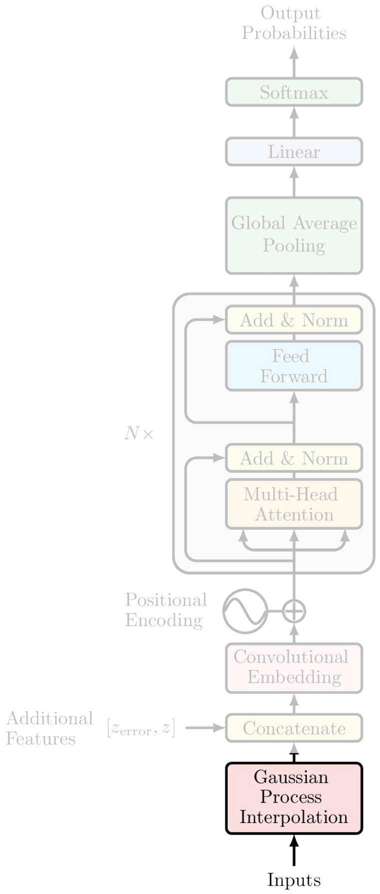


Dealing with Irregularly Sampled Multivariate Time-Series Data

Gaussian Process Interpolation

- 2D-Matern kernel for Gaussian Process interpolation
- Evaluate at regular period, 100 points in our case

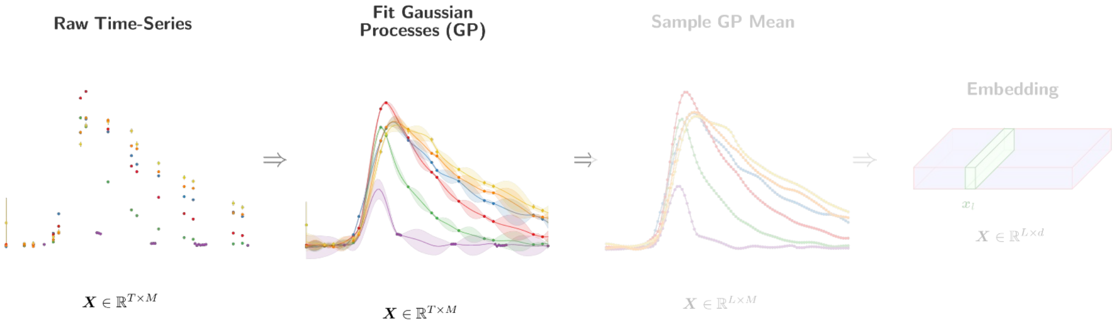


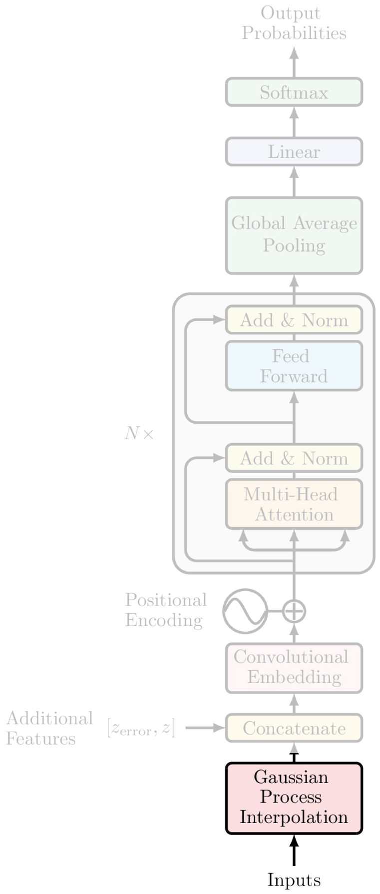


Dealing with Irregularly Sampled Multivariate Time-Series Data

Gaussian Process Interpolation

- 2D-Matern kernel for Gaussian Process interpolation
- Evaluate at regular period, 100 points in our case

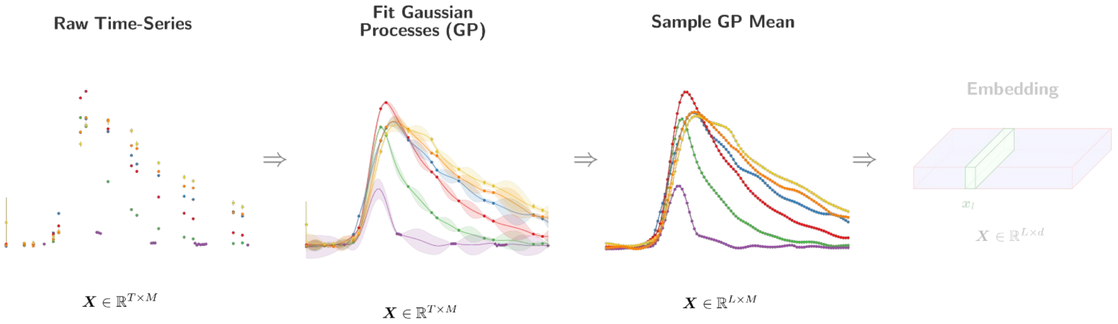


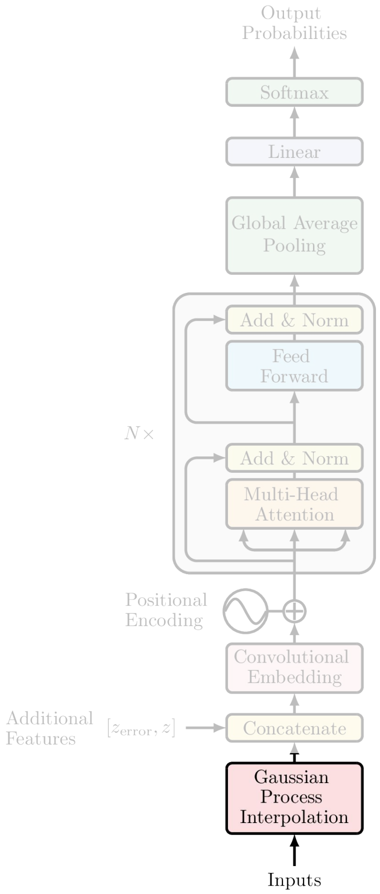


Dealing with Irregularly Sampled Multivariate Time-Series Data

Gaussian Process Interpolation

- 2D-Matern kernel for Gaussian Process interpolation
- Evaluate at regular period, 100 points in our case

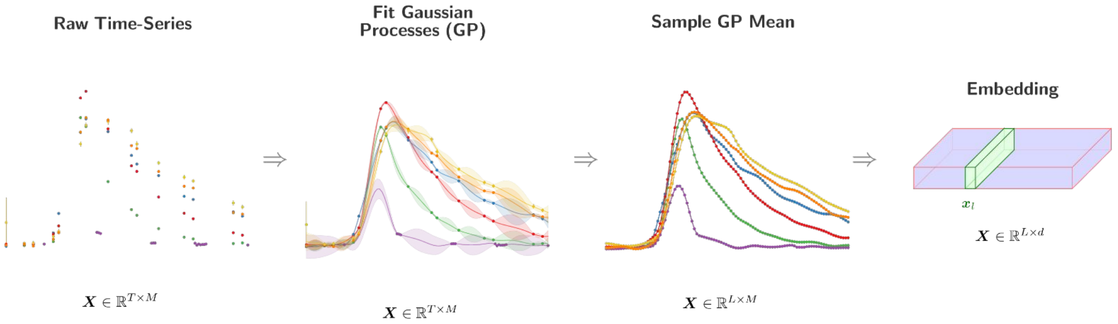


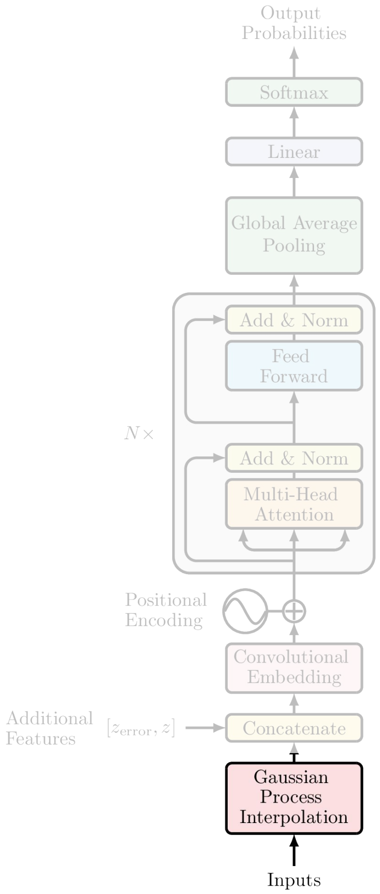


Dealing with Irregularly Sampled Multivariate Time-Series Data

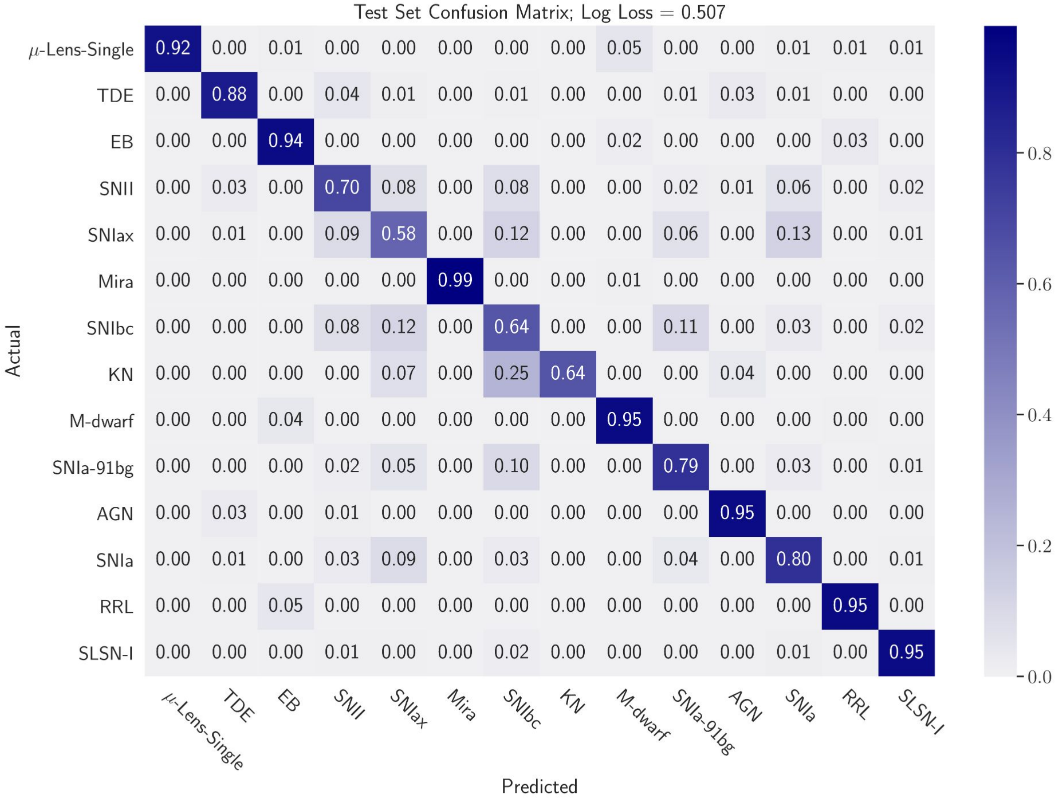
Gaussian Process Interpolation

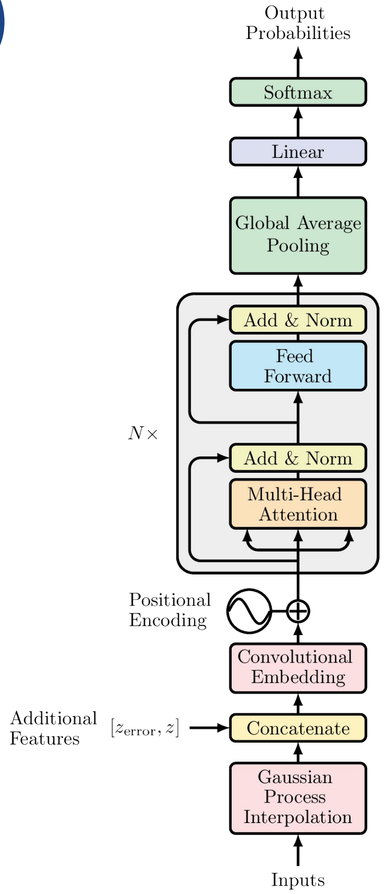
- 2D-Matern kernel for Gaussian Process interpolation
- Evaluate at regular period, 100 points in our case



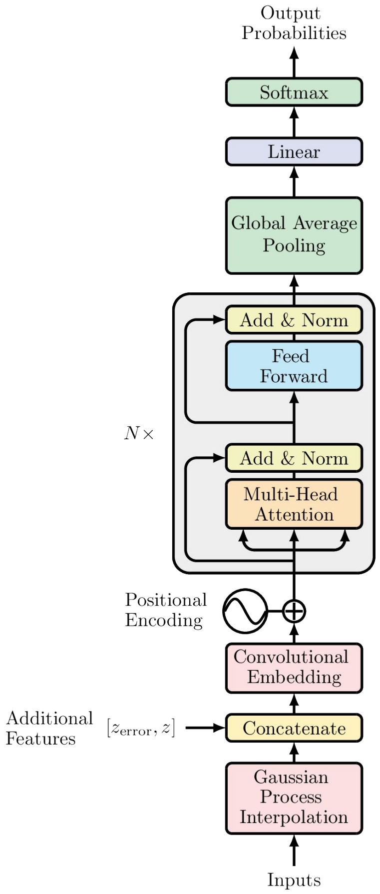


Performance and Results (*ugrizy+Z*)





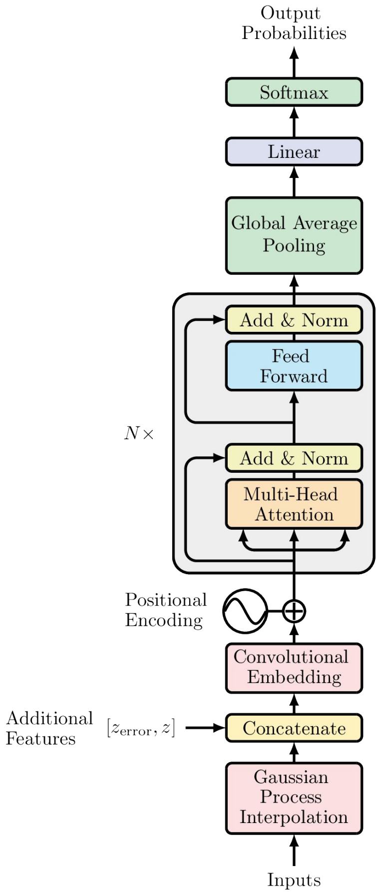
Time for T2 in FINK



Time for T2 in FINK

Considerations

- ZTF Alert stream, only *g* & *r* passbands vs LSST *ugrizy*
- Will only be raw-time series at this point (i.e. no-Z)



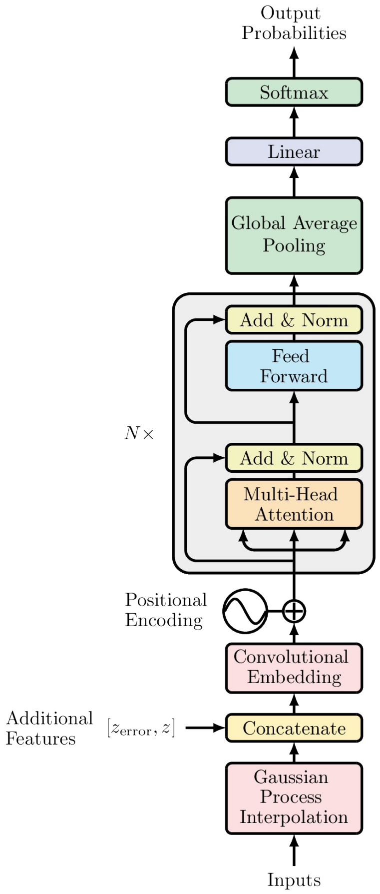
Time for T2 in FINK

Considerations

- ZTF Alert stream, only *g* & *r* passbands vs LSST *ugrizy*
- Will only be raw-time series at this point (i.e. no-Z)

Approaching Implementation

- Stage One: Process Alerts
- Stage Two: Load TF Model for Inference



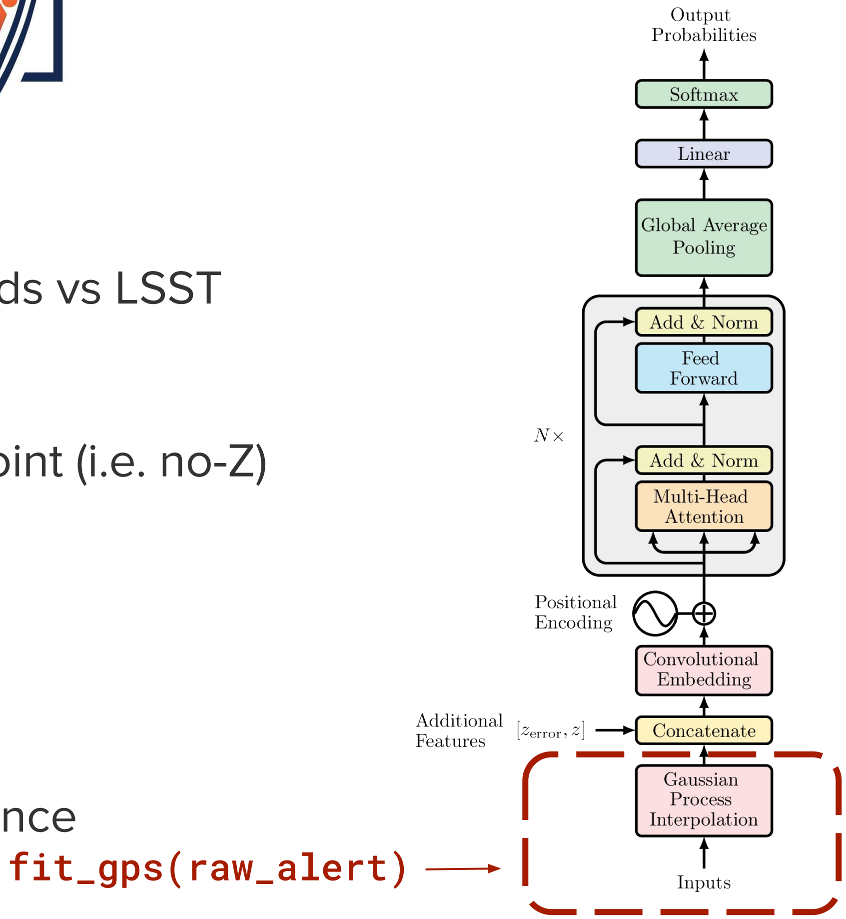
Time for T2 in FINK

Considerations

- ZTF Alert stream, only *g* & *r* passbands vs LSST *ugrizy*
- Will only be raw-time series at this point (i.e. no-Z)

Approaching Implementation

- Stage One: Process Alerts
- Stage Two: Load TF Model for Inference



Time for T2 in FINK

Considerations

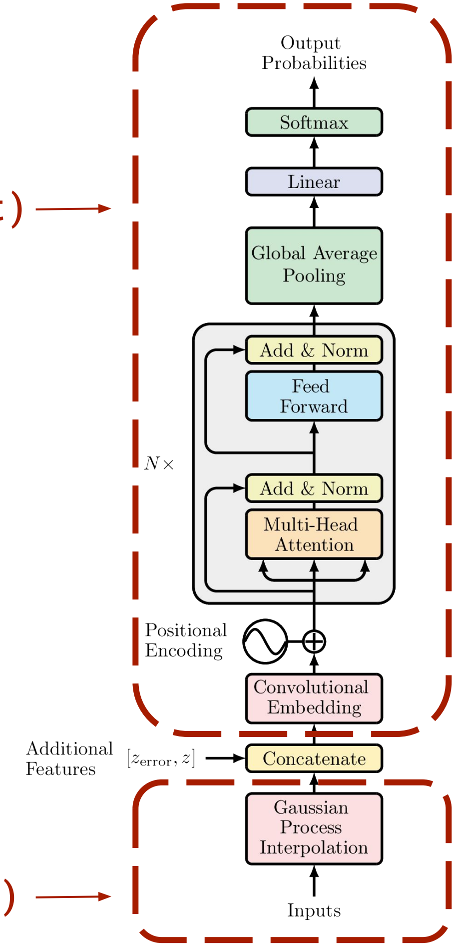
- ZTF Alert stream, only *g* & *r* passbands vs LSST *ugrizy*
- Will only be raw-time series at this point (i.e. no-Z)

Approaching Implementation

- Stage One: Process Alerts
- Stage Two: Load TF Model for Inference

`model.predict(processed_alert)`

`fit_gps(raw_alert)`

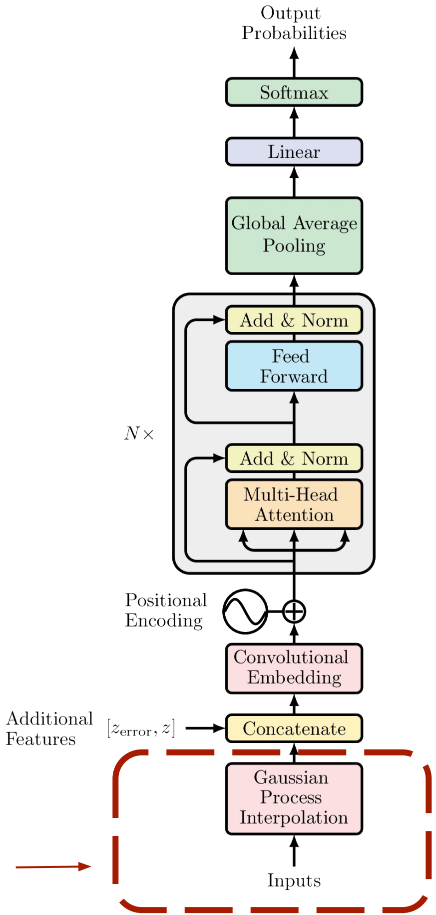


Stage One: Process Alert

Re-Format Raw Alert

```
>>> import pyspark.pandas as ps
>>> psdf = ps.read_parquet('sample.parquet')
>>> import random
>>> r = random.randint(0,len(psdf))
>>> alert = psdf.iloc[r]
>>> print(alert.head())
candid                                          1786552611115010001
schemavsn                                                       3.3
publisher                                                      Fink
objectId                                               ZTF18aaqfhlj
candidate     (2459541.0526157, 2, 1786552611115, 19.1966800...
Name: 221, dtype: object
>>> alert = alert.to_dict()

>>> from fink_client.visualisation import extract_field
# Get flux and error
>>> magpsf = extract_field(alert, 'magpsf')
>>> sigmapsf = extract_field(alert, 'sigmapsf')

>>> jd = extract_field(alert, "jd")

# For rescaling dates to start at 0 --> 30
# dates = np.array([jd[0] - i for i in jd])

# FINK candidate ID (int64)
>>> candid = alert["candid"]

# filter bands
>>> fid = extract_field(alert, "fid")
```

`fit_gps(raw_alert)` →



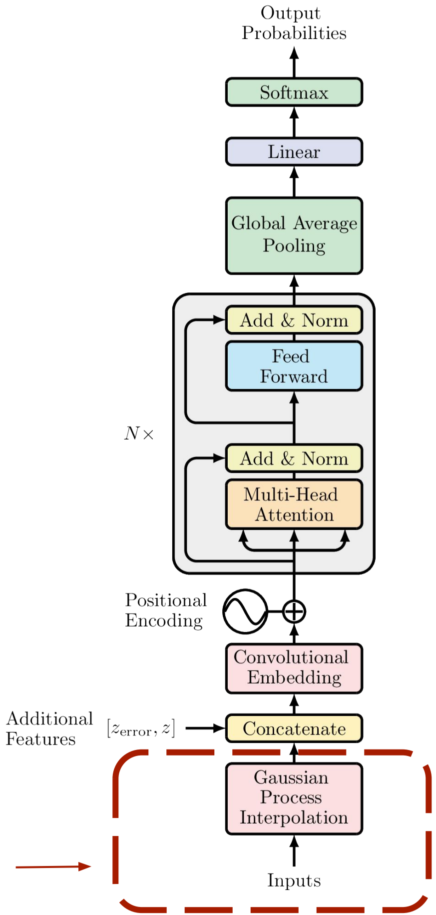
Stage One: Process Alert

Re-Format Raw Alert

```
>>> import pyspark.pandas as ps
>>> psdf = ps.read_parquet('sample.parquet')
>>> import random
>>> r = random.randint(0,len(psdf))
>>> alert = psdf.iloc[r]
>>> print(alert.head())
candid                                        1786552611115010001
schemavsn                                                     3.3
publisher                                                    Fink
objectId                                             ZTF18aaqfhlj
candidate     (2459541.0526157, 2, 1786552611115, 19.1966800...
Name: 221, dtype: object
>>> alert = alert.to_dict()

>>> from fink_client.visualisation import extract_field
# Get flux and error
>>> magpsf = extract_field(alert, 'magpsf')
>>> sigmapsf = extract_field(alert, 'sigmapsf')

>>> jd = extract_field(alert, "jd")

# For rescaling dates to start at 0 --> 30
# dates = np.array([jd[0] - i for i in jd])

# FINK candidate ID (int64)
>>> candid = alert["candid"]

# filter bands
>>> fid = extract_field(alert, "fid")
```

object_id	mjd	flux	flux_error	filter
ZTF18abjrdau	27.9536	16.3509	0.101793	ztfg
ZTF18abjrdau	18.0079	16.7492	0.072451	ztfg
ZTF18abjrdau	16.0102	16.2887	0.09333	ztfg
ZTF18abjrdau	13.9637	16.3082	0.082149	ztfg
ZTF18abjrdau	8.95426	16.4864	0.067591	ztfg

`fit_gps(raw_alert)` →



Stage One: Process Alert

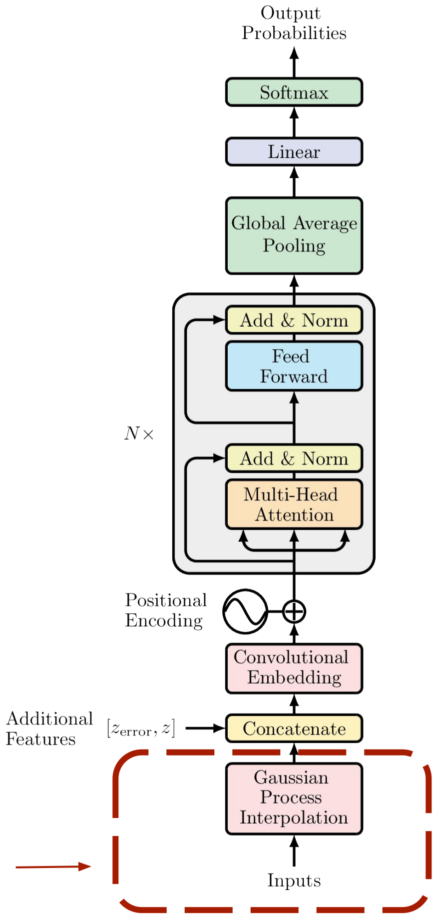
Re-Format Raw Alert

```
>>> import pyspark.pandas as ps
>>> psdf = ps.read_parquet('sample.parquet')
>>> import random
>>> r = random.randint(0,len(psdf))
>>> alert = psdf.iloc[r]
>>> print(alert.head())
candid                                  1786552611115010001
schemavsn                                               3.3
publisher                                              Fink
objectId                                       ZTF18aaqfhlj
candidate     (2459541.0526157, 2, 1786552611115, 19.1966800...
Name: 221, dtype: object
>>> alert = alert.to_dict()

>>> from fink_client.visualisation import extract_field
# Get flux and error
>>> magpsf = extract_field(alert, 'magpsf')
>>> sigmapsf = extract_field(alert, 'sigmapsf')

>>> jd = extract_field(alert, "jd")

# For rescaling dates to start at 0 --> 30
# dates = np.array([jd[0] - i for i in jd])

# FINK candidate ID (int64)
>>> candid = alert["candid"]

# filter bands
>>> fid = extract_field(alert, "fid")
```

+

object_id	mjd	flux	flux_error	filter
ZTF18abjrdau	27.9536	16.3509	0.101793	ztfg
ZTF18abjrdau	18.0079	16.7492	0.072451	ztfg
ZTF18abjrdau	16.0102	16.2887	0.09333	ztfg
ZTF18abjrdau	13.9637	16.3082	0.082149	ztfg
ZTF18abjrdau	8.95426	16.4864	0.067591	ztfg

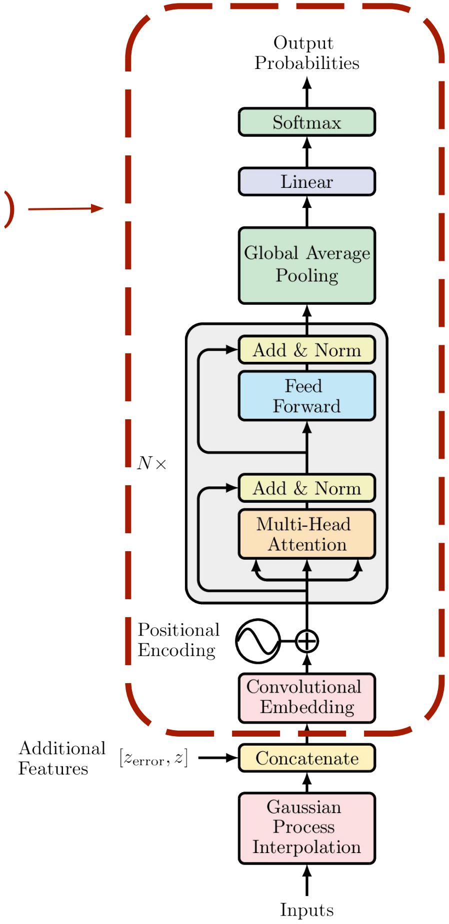
`fit_gps(raw_alert)` →



Stage Two: Black Box Inference

Load Pre-trained Model

`model.predict(processed_alert)`

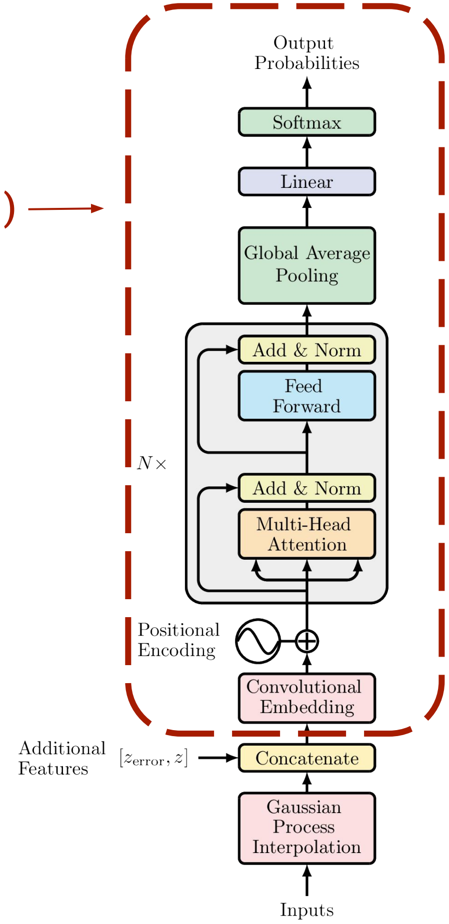


Stage Two: Black Box Inference

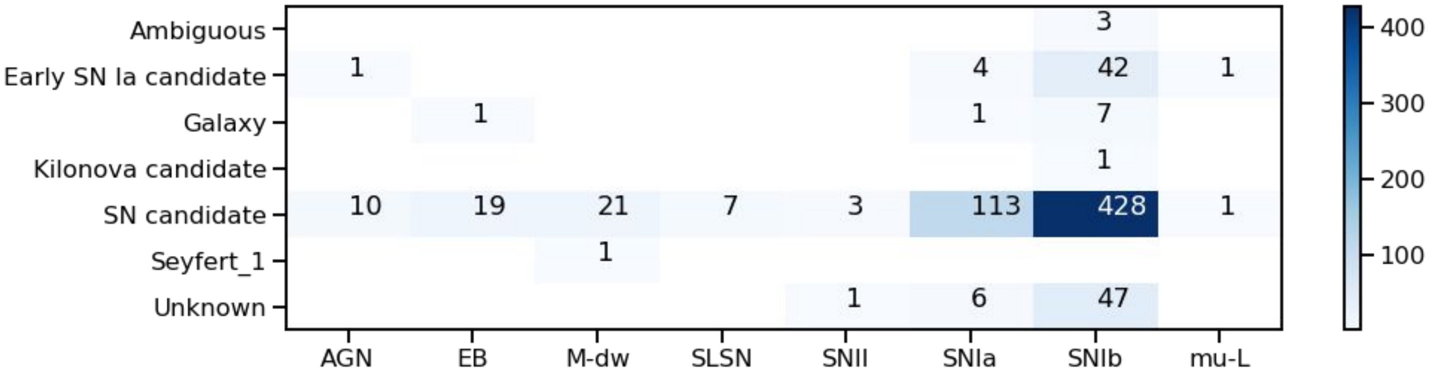
Load Pre-trained Model

```
def get_model(model_name: str = 't2', model_id: str = "23057-1642540624-0.1.dev963+g309c9d8"):
    """ Load pre-trained model for T2

    Parameters
    ----------
    model_name: str
        Folder name containing pre-trained models. Available: t2, atx
    model_id: str
        Corresponding ID inside the foler (related to the version used to train)

    Returns
    ----------
    out: keras model
    """
    model_path = (
        f"{Path(__name__).absolute().parent.parent}/data/models/{model_name}/model-{model_id}"
    )

    model = keras.models.load_model(
        model_path,
        custom_objects={"WeightedLogLoss": WeightedLogLoss()},
        compile=False,
    )

    return model
```

`model.predict(processed_alert)` →



Performance, *thus far …*

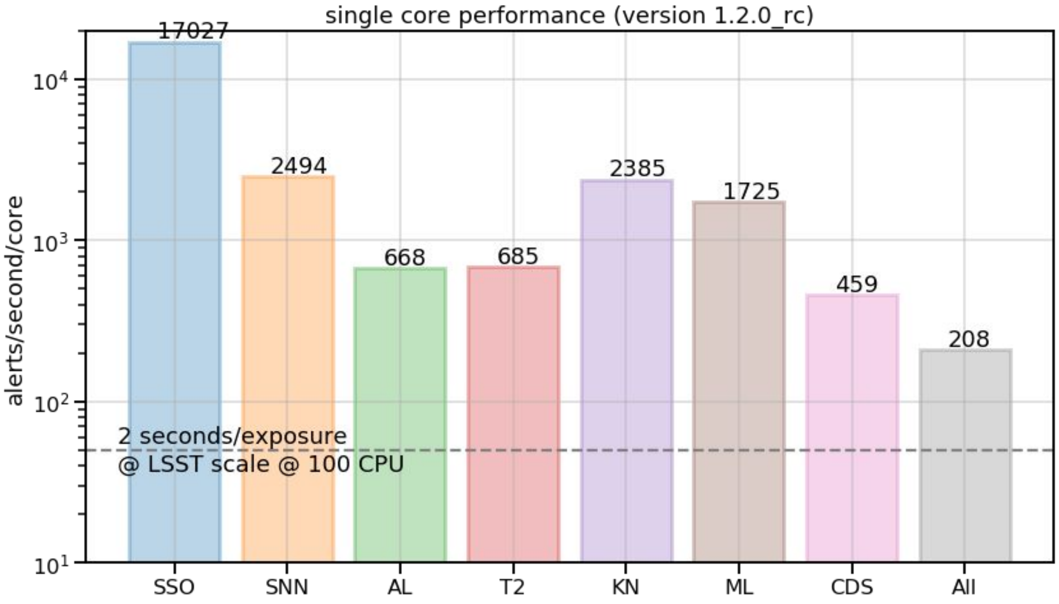


Trial Run:

- Selection cuts: Only 2 filters, at least 2 points per band, max history 90 days
- 161,462 –> 715 alerts (one full night) after selection cuts, compare "max prob" class from t2 with FINK scoring.
- New predictions for "Unknown", but seems to be a bias for SNIb.
- TNS validation does well for SN in general, but confirms a slight bias to SNIb instead of SNIa.

Performance, ***thus far …***

- Still a few tweaks that could improve throughput
- Need to better understand t2's *value add,* max_prob vs all probs?
- Where should t2 sit along the pipeline?

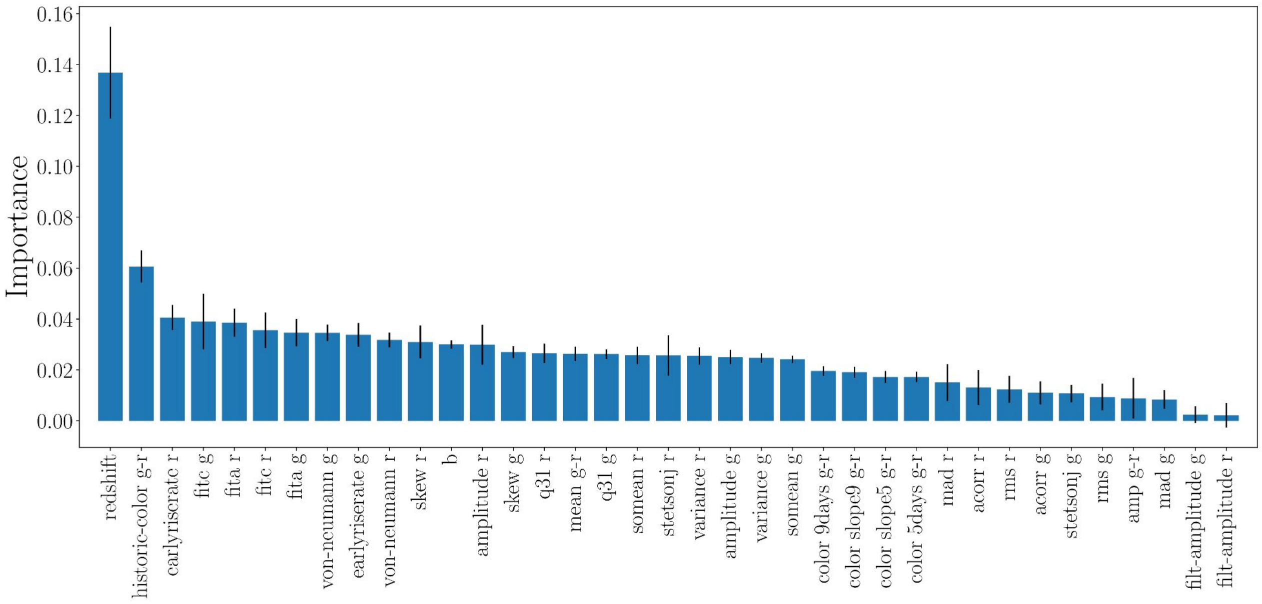


Discussion…

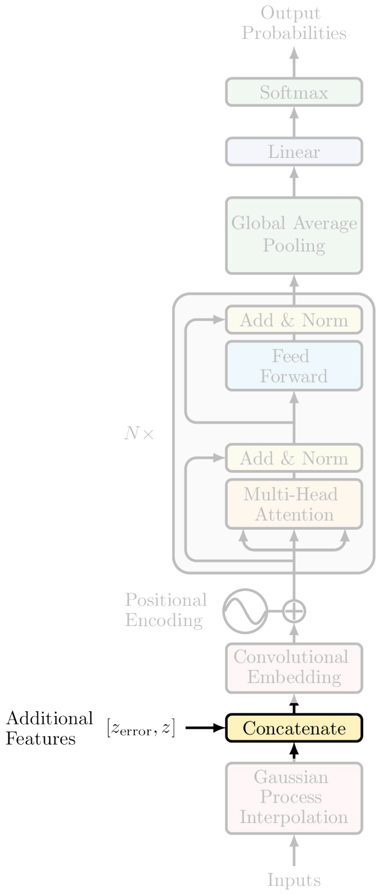
APPENDIX

Inputting Addition Information

- Domain knowledge still important!



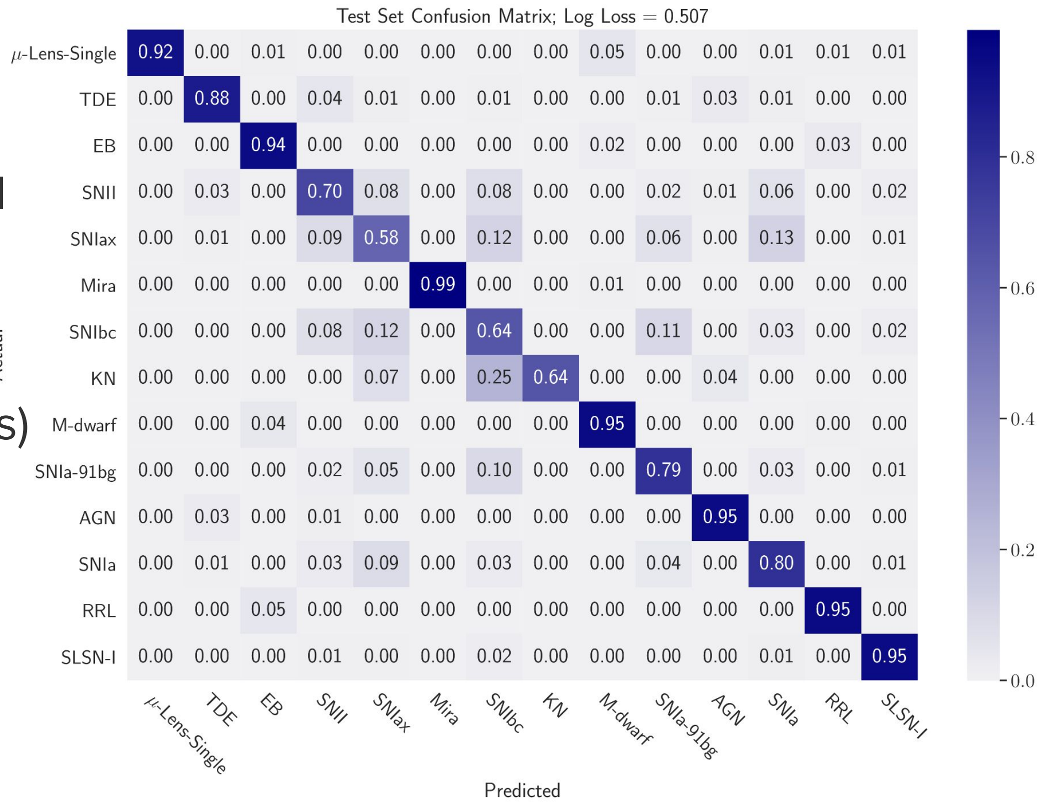
*Boone et al. 2019



Performance and Results (PLAsTiCC:*ugrizy+Z*)

Confusion Matrix

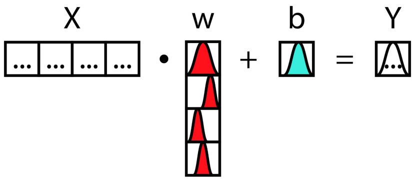
- Handles class imbalance well
- Log-Loss **0.507** (with *z*)
- Log-Loss **0.8** (only time-series)
- Should be noted this is in a *representative* setting



Extensions and Future Work

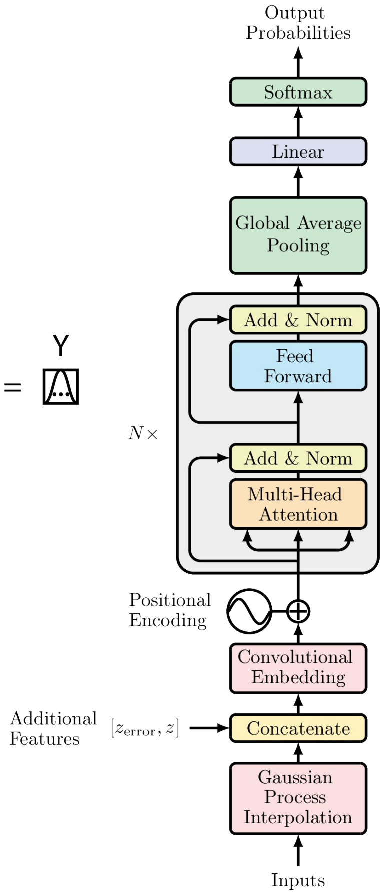
Extending Time-Series Transformer [t2]

- Inclusion of Decoder-block for early light curve classification
- Probabilistic extensions for better UQ



Future Work

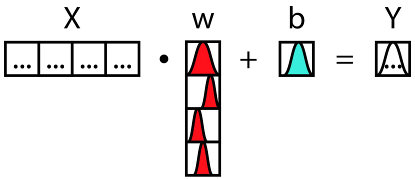
- Test on real data
- Put into production in real-time setting

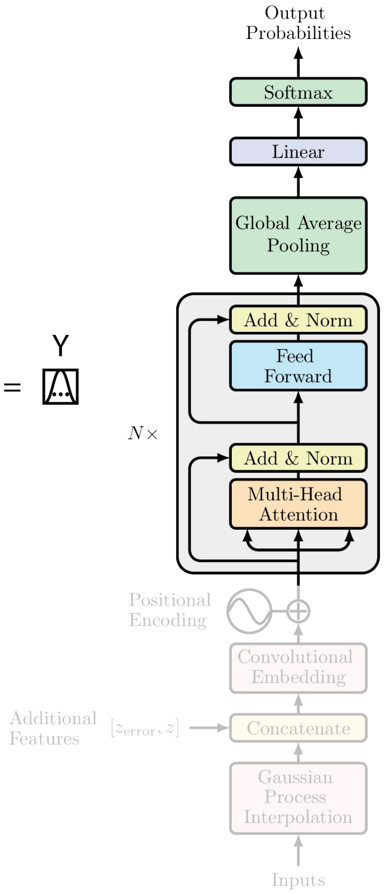


Extensions and Future Work

Extending Time-Series Transformer [t2]

- Inclusion of Decoder-block for early light curve classification
- Probabilistic extensions for better UQ





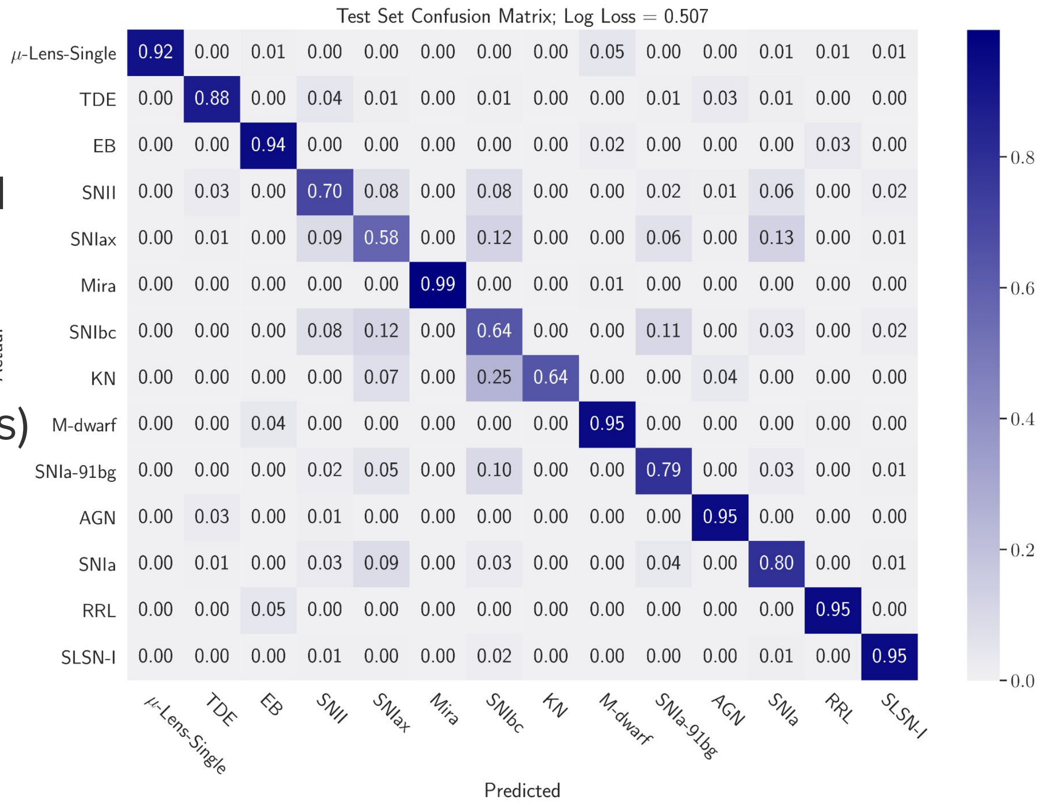
Future Work

- Test on real data
- Put into production in real-time setting

Performance and Results

Confusion Matrix

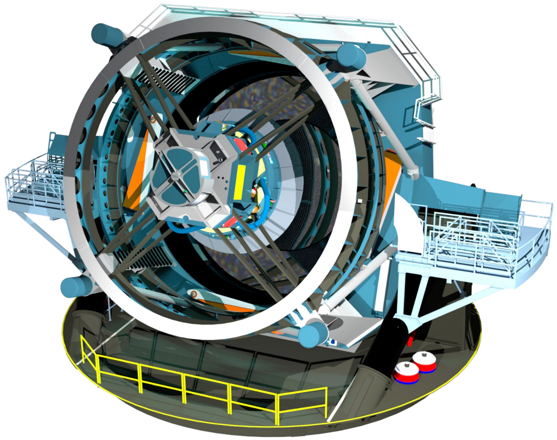
- Handles class imbalance well
- Log-Loss **0.507** (with *z*)
- Log-Loss **0.8** (only time-series)
- Should be noted this is in a *representative* setting



Motivation: A Deluge of Data

Overview

- 10 million alerts, per night!
- Machine Learning methods are now critical
- Accurate and fast classification required for follow up
- Desire for UQ and interpretability when allocating follow up resources.



Deep Learning to the Rescue!?

The death of feature engineering?

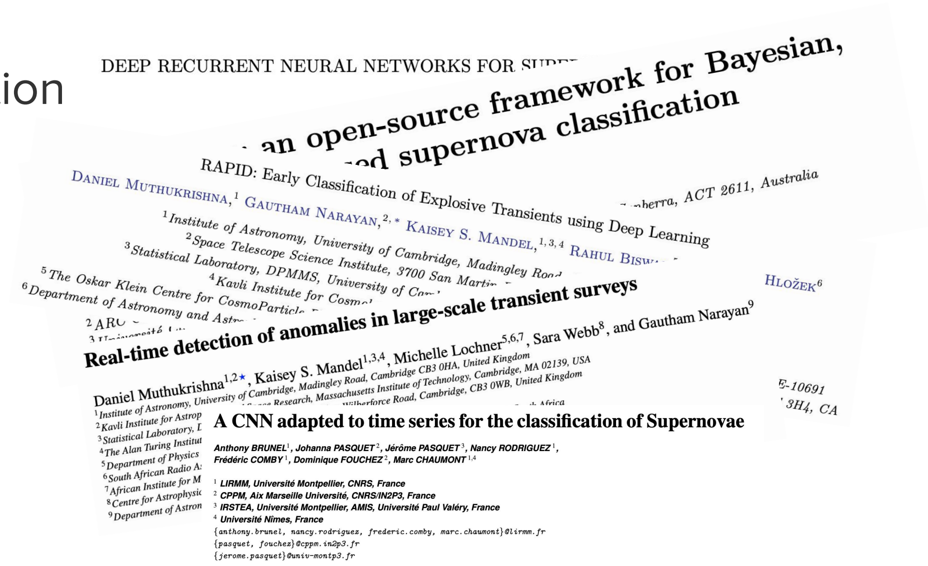
- Exploiting inherent time-series information

RNNs (inc. LSTMs, GRUs)

- Hard to train, not easily parallelizable
- Large memory footprint

CNNs (inc. TCN)

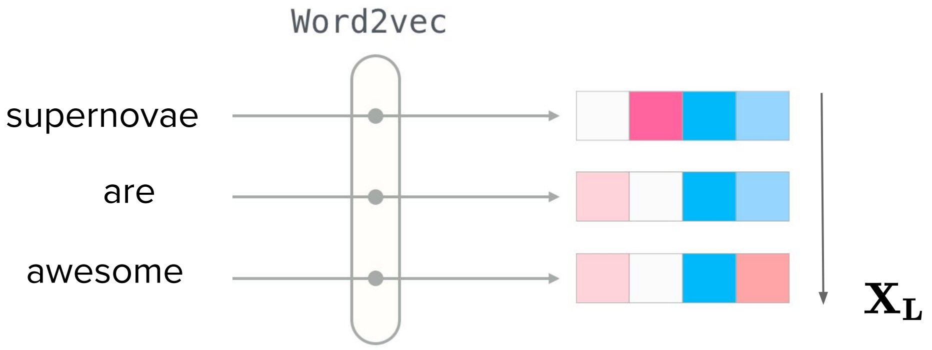
- Computationally expensive



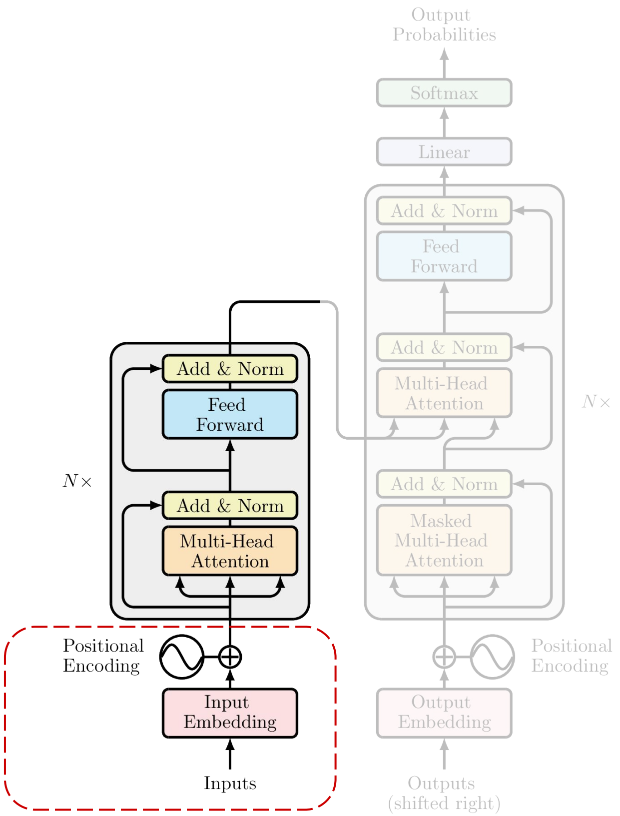
Input Embedding and Positional Encoding

Vector space representations

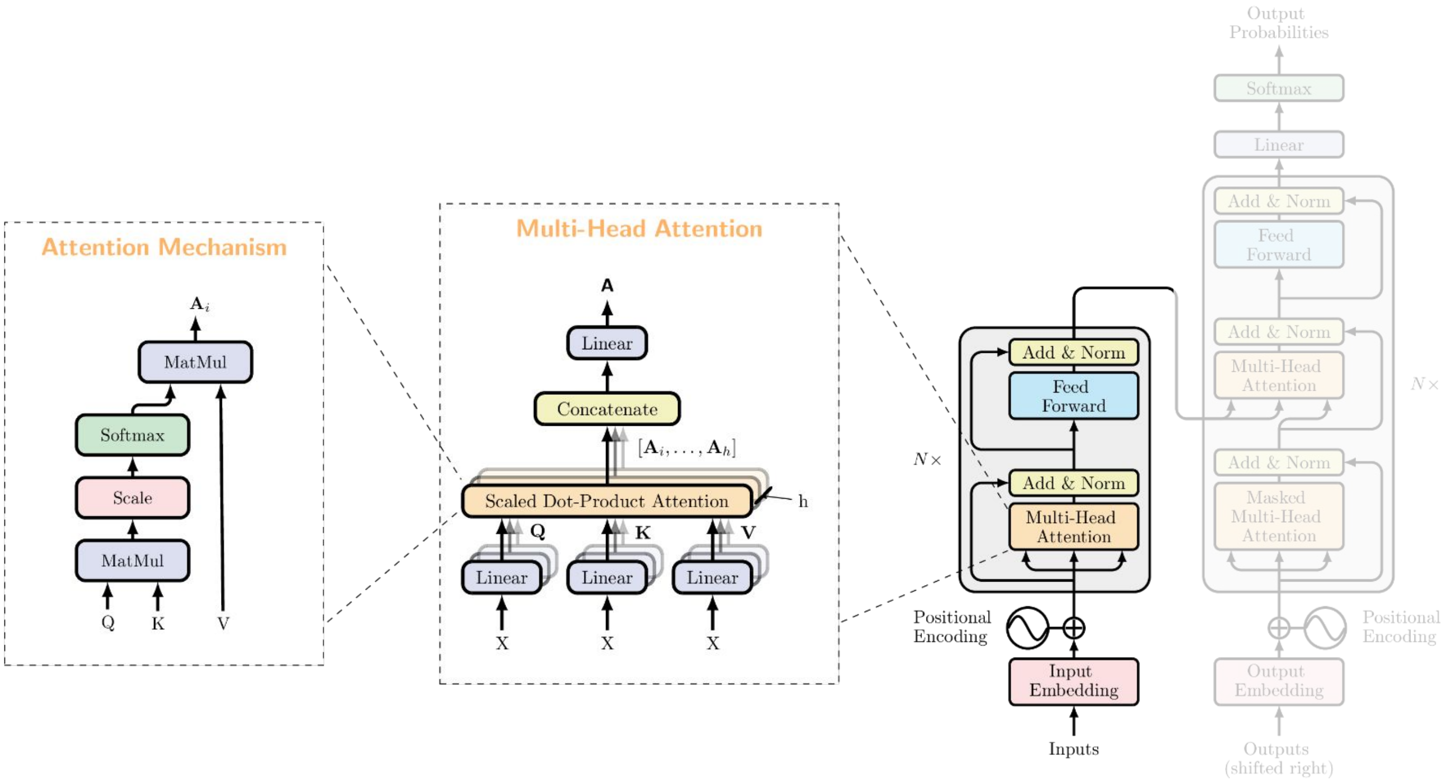
"supernovae are awesome"



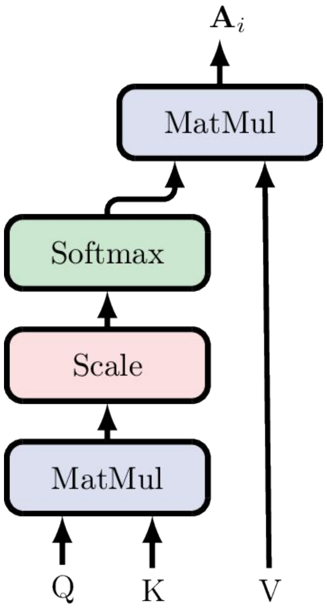
$$\boldsymbol{x}_l \leftarrow \boldsymbol{x}_l \oplus \boldsymbol{p}_l.$$



Multi-headed Self Attention



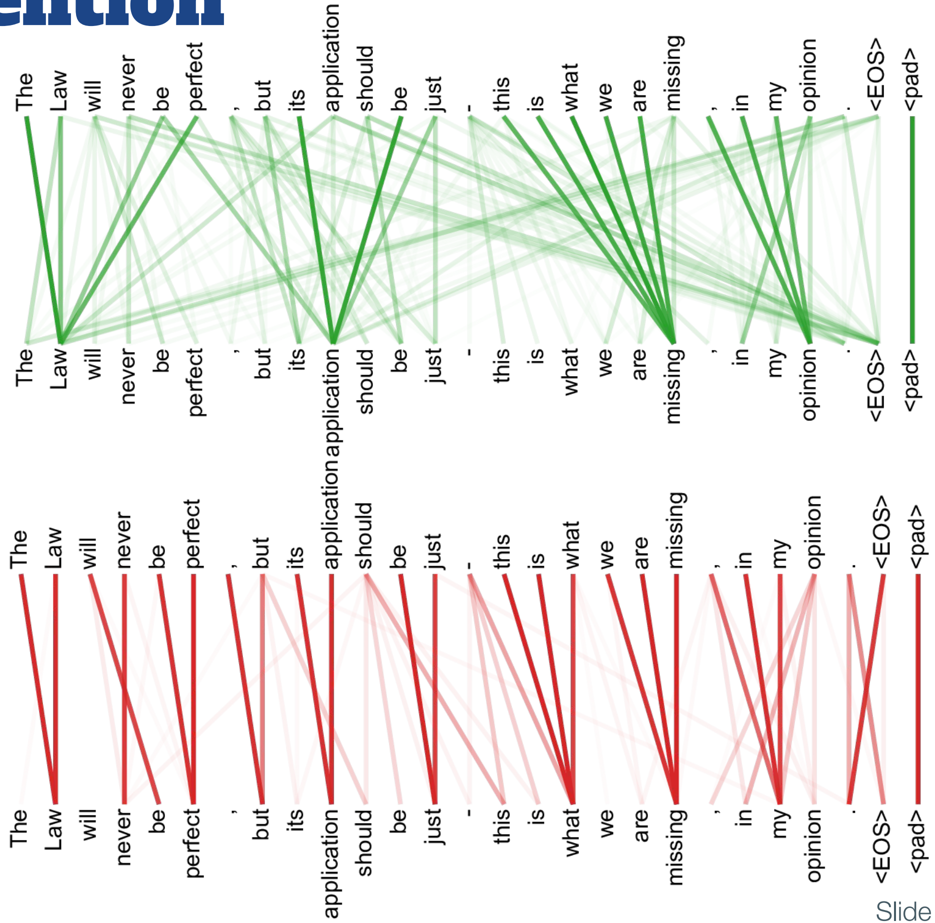
Multi-headed Self Attention



$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathbf{A} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V}.$$

$$\mathbf{Q} = \mathbf{X}\mathbf{W}^Q \in \mathbb{R}^{L \times d_q},$$

$$\mathbf{K} = \mathbf{X}\mathbf{W}^K \in \mathbb{R}^{L \times d_k}, \mathbf{V} = \mathbf{X}\mathbf{W}^V \in \mathbb{R}^{L \times d_v}$$



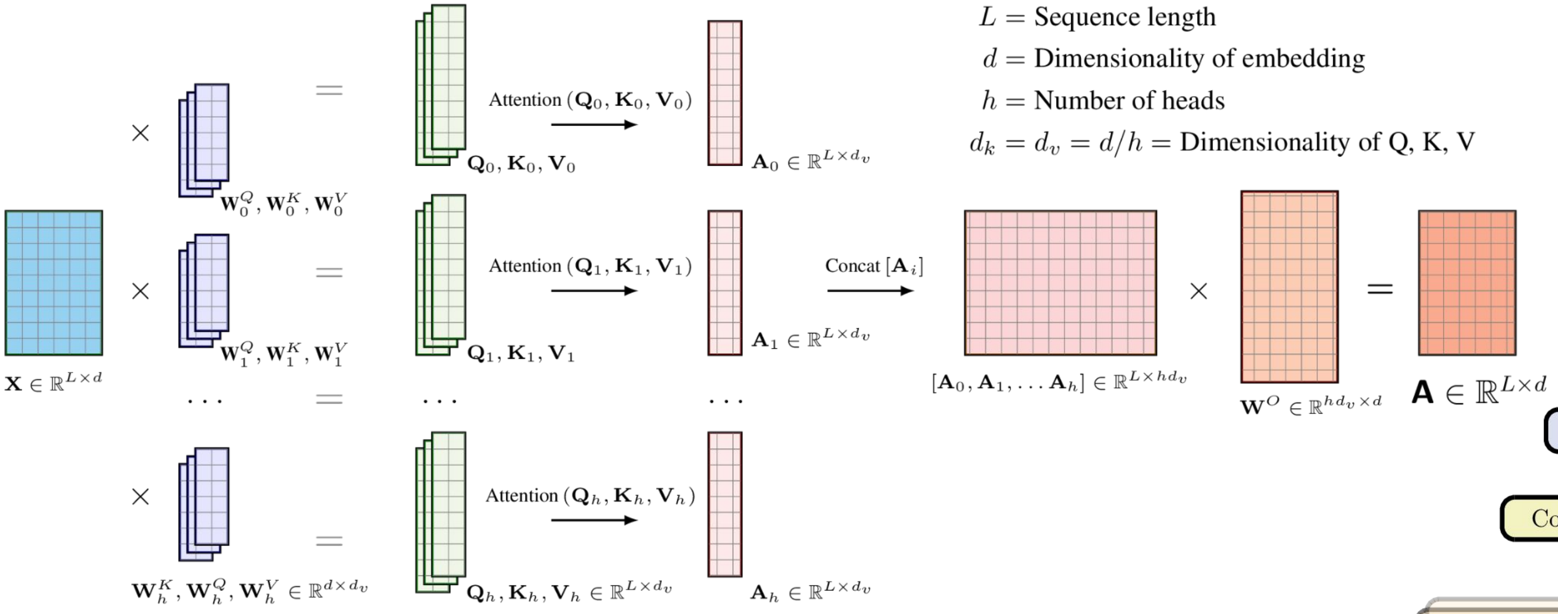
Multi-headed Self Attention

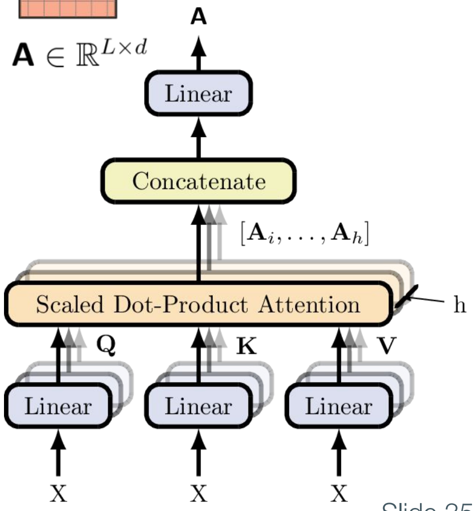
L = Sequence length

d = Dimensionality of embedding

h = Number of heads

$d_k = d_v = d/h$ = Dimensionality of Q, K, V



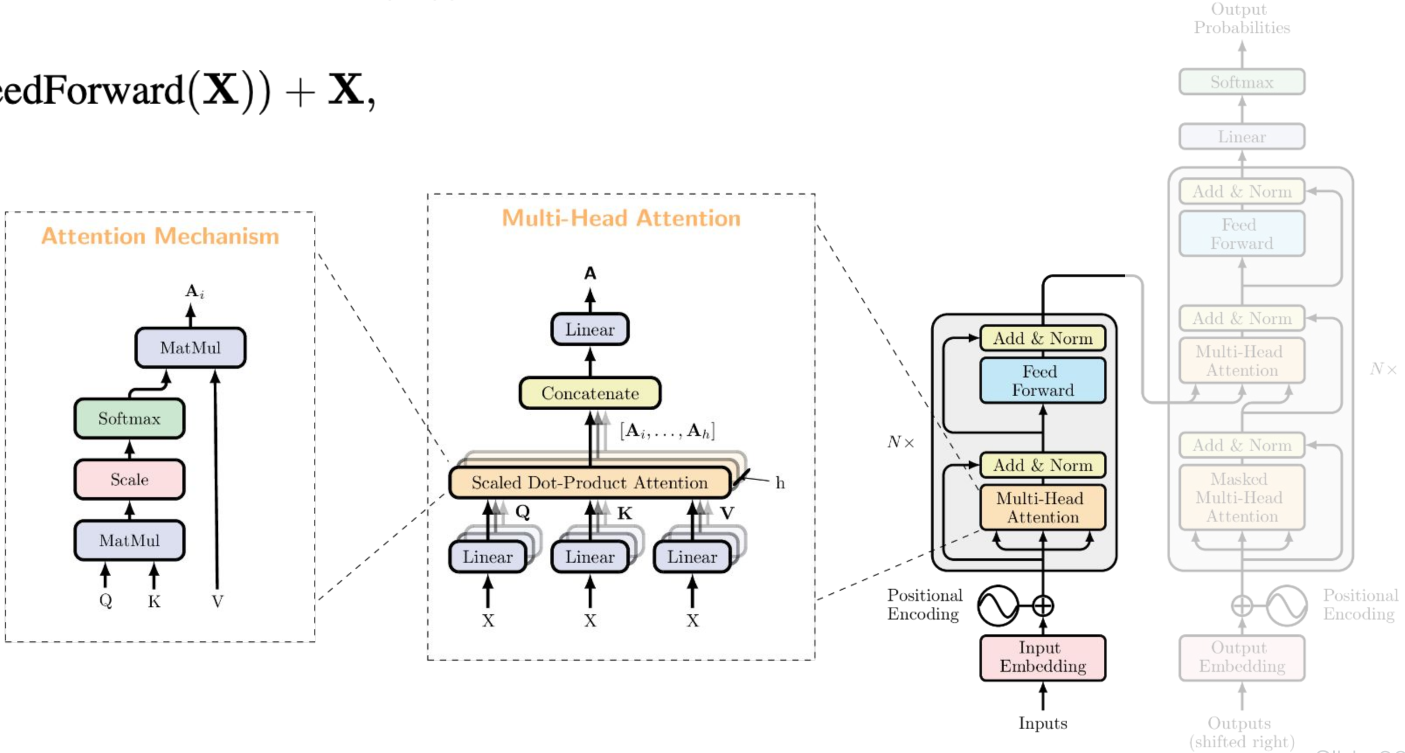


$$\text{MultiHead}\left(\mathbf{Q}, \mathbf{K}, \mathbf{V}\right) = \mathbf{A} = \text{Concat}[\mathbf{A}_1, \ldots, \mathbf{A}_h]\mathbf{W}^O,$$

A Transformer Block [Encoder]

1. $\mathbf{X} \leftarrow \text{LayerNorm}(\text{MultiHeadSelfAttention}(\mathbf{X})) + \mathbf{X}.$

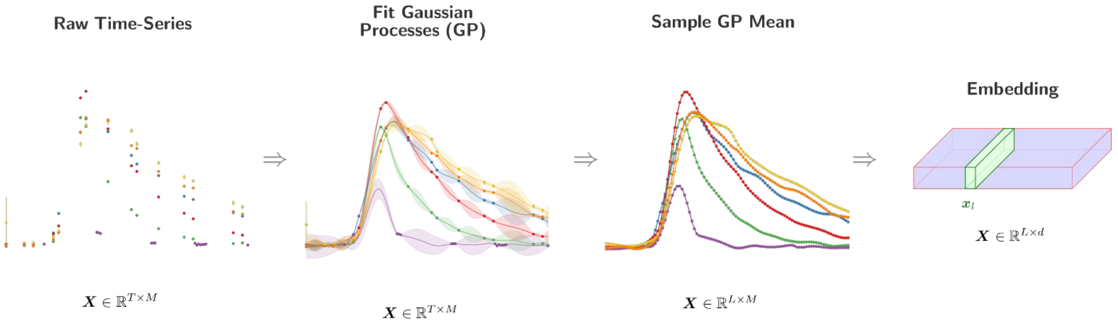
2. $\mathbf{X} \leftarrow \text{LayerNorm}(\text{FeedForward}(\mathbf{X})) + \mathbf{X},$

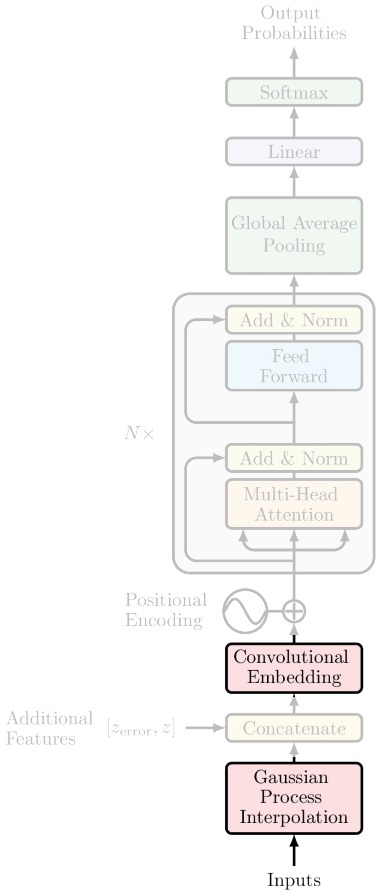


Convolutional Embedding

Projecting photometric data into vector spaces

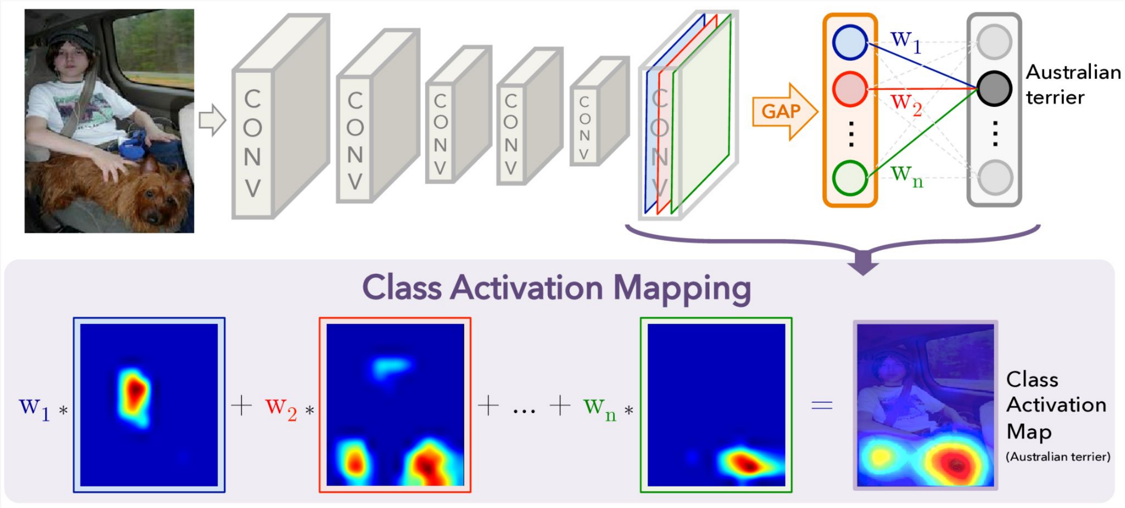
- Think of `timestep2vec` akin to `word2vec`
- 1-dimensional convolution allows for scaling of M → d
- Positional encoding follows as before

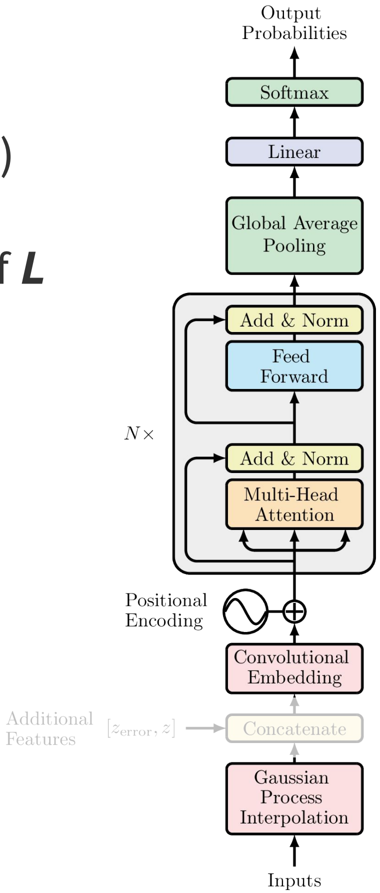




Global Average Pooling (GAP)

- Inclusion of GAP allows for Class Activation Maps (CAMs)
- CAMs adapted for use with time-series, i.e. as function of ***L***

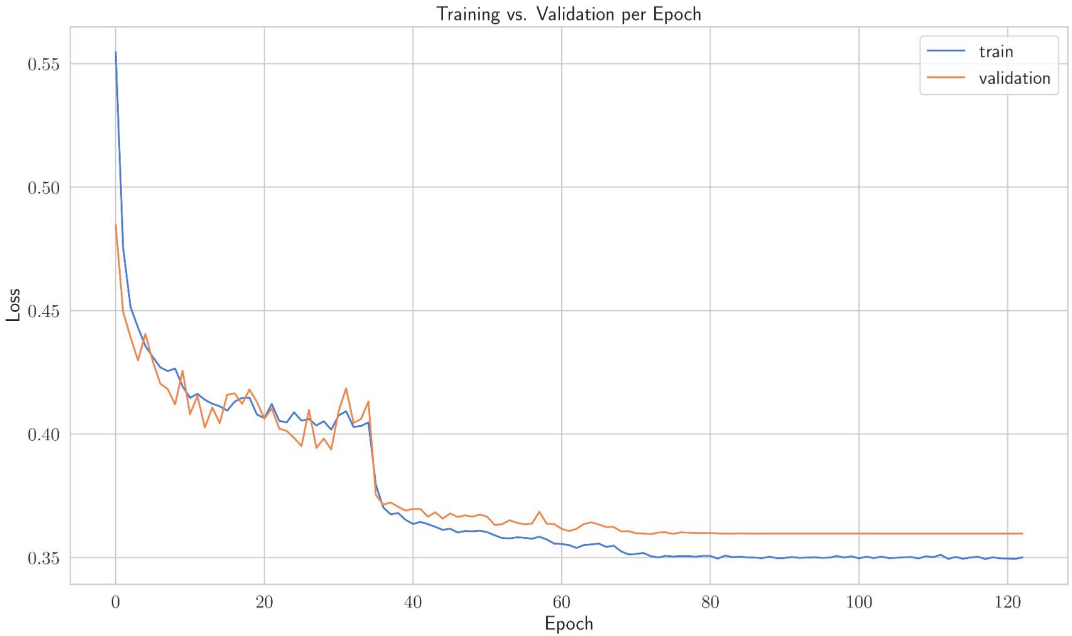




Performance and Results

Training and Inference

- PLAsTiCC dataset ~ 3M LCs
- Imbalanced, representative
- Extensibility for multi-GPU or TPU
- Low #params for fast inference*

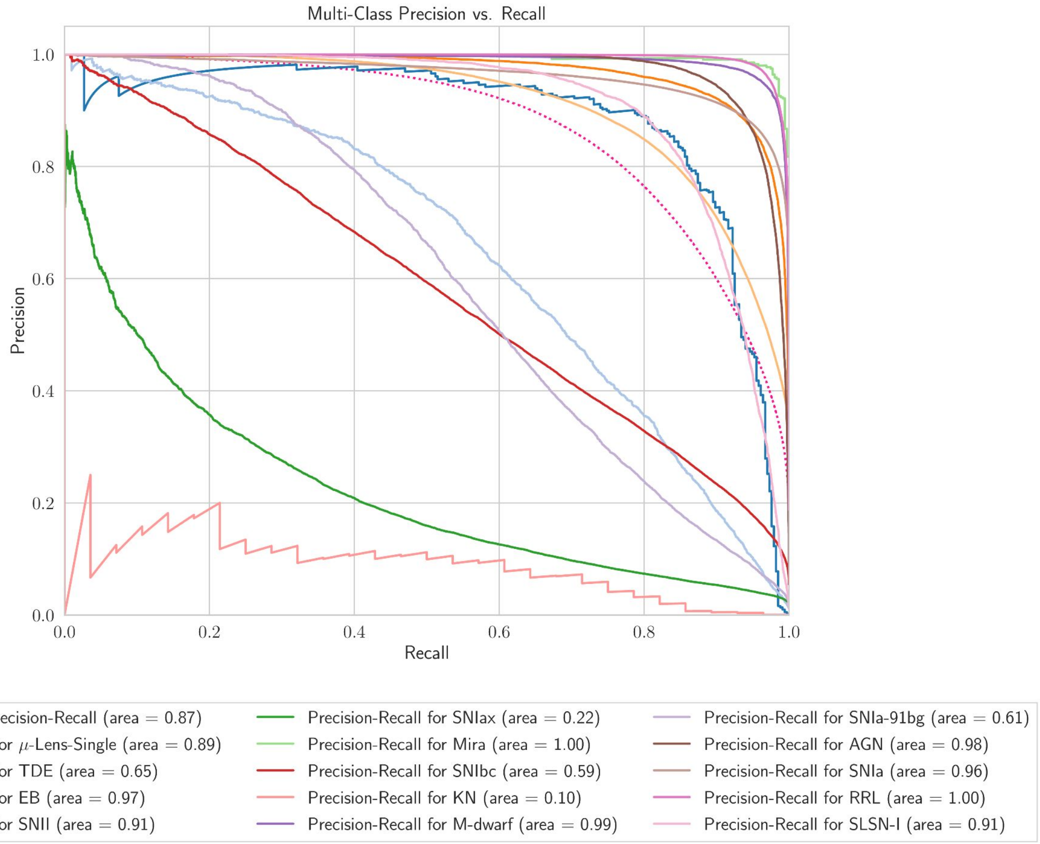


**does not include GP interpolation step*

Performance and Results

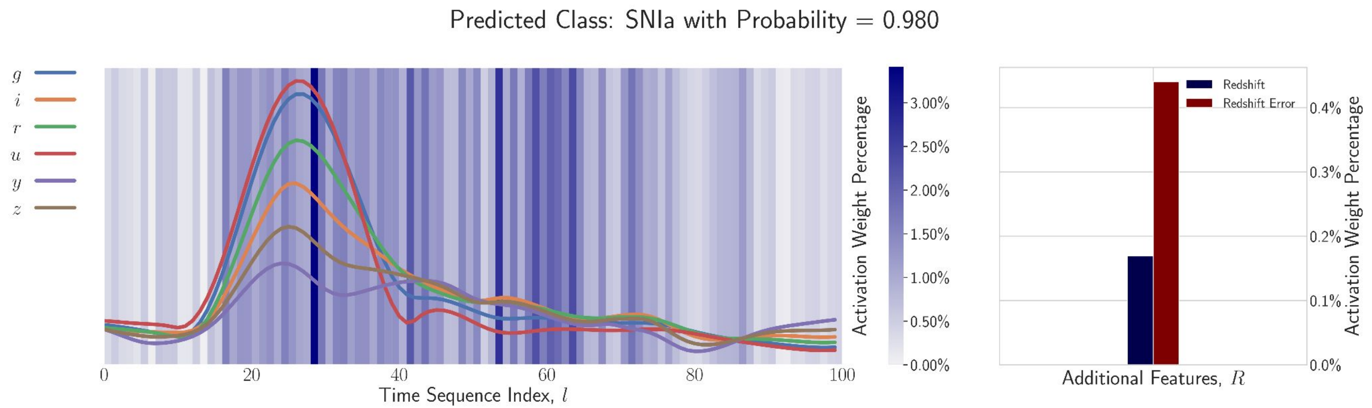
Precision-Recall

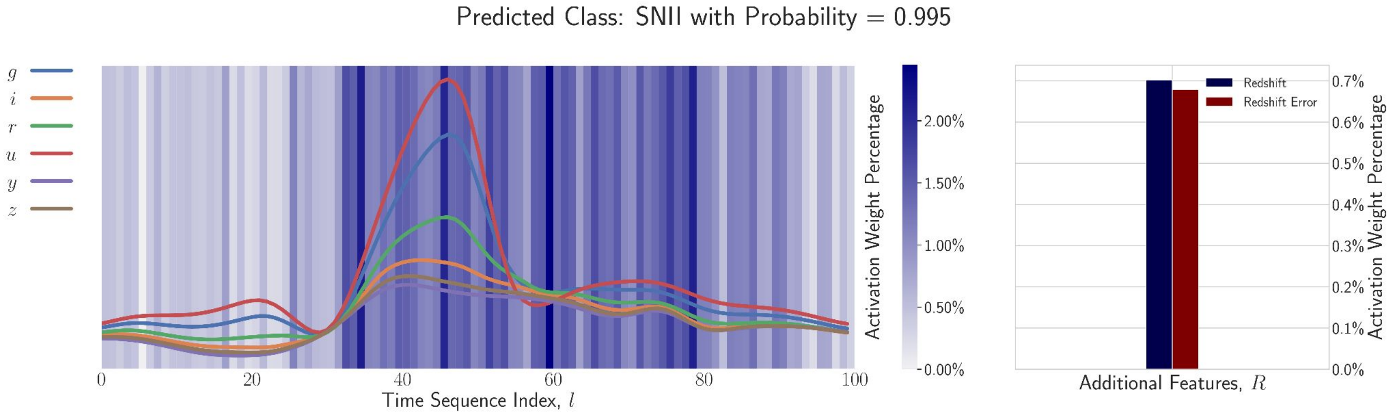
- AUC ~ **0.87**
- KNe struggling (0.004%)
- SNIax mistaken for Type Ia?



Performance and Results

CAMs





Questions?