Universidad de Concepción

Dirección de Postgrado

Facultad de Ingeniería - Programa de Magíster en Ciencias de la Computación

POSITIONAL ENCODINGS FOR LIGHT CURVE TRANSFORMERS: AN EVALUATION OF THEIR IMPACT IN THE PRETRAINING AND CLASSIFICATION TASK

Tesis para optar al grado de

MAGÍSTER EN CIENCIAS DE LA COMPUTACIÓN

POR

DANIEL ANDRÉS MORENO CARTAGENA

CONCEPCIÓN, CHILE

Marzo, 2024

Profesor guía: GUILLERMO CABRERA VIVES

Departamento de Ingeniería Informática y Ciencias de la Computación

Facultad de Ingeniería

Universidad de Concepción

Abstract

The vast volume of astronomical data generated nightly by observatories, such as the Vera C. Rubin Observatory, presents significant challenges in the classification and analysis of light curves. These curves, characterized by their unique distributions across various bands, irregular sampling, and varied cadences, necessitate sophisticated models capable of generalization across diverse astronomical surveys. In this work, we conducted empirical experiments to assess the transferability of a light curve transformer model to datasets with different cadences and magnitude distributions, utilizing various positional encodings. We proposed a new approach to directly incorporate temporal information into the output of the last attention layer. Additionally, we modified the common finetuning approach to assess the adaptability of the light curve transformer in contexts where the cadence is markedly different from that of the dataset used for its pretraining. Our results indicate that using trainable positional encodings leads to significant improvements in transformer performance and training times. Our proposed positional encoding, applied to the attention mechanism, can be trained more quickly than the traditional non-trainable positional encoding transformer, while still achieving competitive results when transferred to other datasets. Our approach to adapting the model to a dataset with a very different cadence demonstrates that, in terms of reconstruction of astronomical time series, both training time and computational space can be reduced. This approach achieves an adaptation in the cadence of the survey without the need to train the entire model, indicating a promising direction for future research in astronomical data analysis.

Contents

List of Figures

Chapter 1

Introduction

The Vera C. Rubin Observatory (LSST; Ivezić et al. 2019) will produce a vast number of observations every night in the sky, reaching up to 40 million events per night where the brightness or location of a source change (Sánchez-Sáez et al. 2021). The classification of this data is of utmost importance to astronomers as it allows them to acquire information about the physical characteristics and properties of astronomical objects.

In recent years, the development of deep learning models has aided in categorization of light curves (Charnock and Moss 2017; Donoso-Oliva et al. 2021; Muthukrishna et al. 2019). However, light curves pose a considerable challenge as they have different distributions in each of the bands, are irregularly sampled and have varying cadences depending on the telescope at which measurements were taken (Pasquet et al. 2019; Yu et al. 2021). These peculiarities make it difficult to generate models that are sufficiently generalizable to all astronomical surveys.

Transformers, a type of deep learning model that use self-attention, have demonstrated exceptional performance on light curves (Astorga et al. 2023; Donoso-Oliva et al. 2022; Pimentel et al. 2022). In these models, temporal information is conveyed through positional encoding (PE), typically provided by sine and cosine functions with varying frequencies (Vaswani et al. 2017). However, the original proposal of PE was made in the context of text data, assuming uniform word spacing, which differs from the characteristics of astronomical time series. To tackle this challenge, we have focused on enhancing the robustness of the

PE definition to effectively generalize to different astronomical surveys.

Currently, there is limited research utilizing positional encodings in transformer models to represent temporal information in light curves. Allam Jr and McEwen (2022), Donoso-Oliva et al. (2022), and Morvan et al. (2022) employed non-trainable positional encodings by directly inserting timestamps into the predefined function in Vaswani et al. (2017), achieving encouraging results compared to traditional deep learning methods. Pimentel et al. (2022) and Astorga et al. (2023) proposed a new module based on Fourier decomposition, with M harmonic components and trainable parameters, to induce temporal information. Additionally, Pan et al. (2022) utilized the rotary positional encoding proposed in Su et al. (2021) to explicitly leverage relative positions in the self-attention formulation. However, no studies have analyzed the effect of positional encoding of a transformer model within the light-curve domain.

Motivated by the above, we test different PEs, in a light curve transformer, and evaluate their performance in the reconstruction of astronomical time series (pretraining and finetuning stages) and the classification of variable stars. We use the architecture proposed in Donoso-Oliva et al. (2022), which is a self-supervised light curve transformer model. We perform an empirical comparison and analysis of several PEs on astronomical surveys with different cadences. Additionally, we propose a new approach to incorporate the temporal information and investigate the potential of adapting a transformer model to a dataset with a significantly different cadence, without the need to train the entire model.

The remainder of this work is organized as follows: Firstly, Chapter 2 introduces the fundamental concepts underlying the conducted research. Next, Chapter 3 elaborates on the employed methodology in detail. Subsequently, Chapter 4 presents the analytical and experimental results, discussion and limitations. Lastly, the final conclusions and future research are given in Chapter 5.

1.1 Hypothesis

1. Employing a trainable positional encoding during pretraining yields improved performance in terms of reduced total pretraining time without decrease in

classification performance.

2. A trainable positional encoding within a light curve transformer is capable of capturing sufficiently representative temporal relationships in a specific dataset.

3. To adapt a transformer pretrained on an astronomical dataset with a given cadence to another dataset with a noticeably different cadence, we only need to train the positional encoding.

1.2 Objectives

1.2.1 General Objective

Implement and evaluate trainable positional encodings on a light curve transformer to improve the ability to adapt to astronomical surveys with different cadences.

1.2.2 Specific Objectives

i Implement trainable positional encodings, acquired from the existing literature, within the ASTROMER architecture proposed by Donoso-Oliva et al. (2022).

ii Generate datasets with varying cadences, derived from the astronomical survey on which the transformer was pretrained.

iii Conduct experimental evaluations on different astronomical surveys during both the pretraining and finetuning stages, assessing performance in terms of Root Mean Square Error (RMSE), R-Squared (R^2), and total training time. Additionally, perform experimental evaluations in the classification stage, assessing performance based on the F1-score metric.

iv Analyze, compare, and discuss the effects of positional encoding on the reconstruction and classification of light curves.

Chapter 2

Theoretical Background

This chapter presents the important concepts that support this thesis and is distributed in two main sections: Section 2.1 defines relevant astronomical concepts and Section 2.2 explores necessary concepts, from the point of view of deep learning, to understand the work done.

2.1 Astronomy

In astronomy, spectroscopy and photometry are two different techniques used to measure and analyze the light emitted by celestial objects. Spectroscopy measures light in terms of its wavelength, which allows light to be broken down into its different spectral components, while photometry measures the intensity of light emitted by a celestial object in a specific range of wavelengths, usually in a narrow band of the electromagnetic spectrum.

2.1.1 Spectroscopy

Spectroscopy is the study of the interaction between electromagnetic radiation and matter, with absorption or emission of radiant energy (Kirchhoff and Bunsen, 2008). In astronomy, it is an observational technique oriented to the analysis of the spectral composition of the light received from the celestial objects (Huggins and Miller, 1863). Spectroscopy allows to analyze the composition of that light and deduce how much energy is received from a stars for each wavelength. Astronomers use that wavelength to identify the chemical components of the objects through the absorption lines obtained in the spectrum. In particular, this technique allows to analyze the light emitted or absorbed by celestial objects and obtain detailed information on its temperature, rotation and displacement speed with respect to the observer, among other aspects, as well as concentrations of Hydrogen, Helium and traces of Silicon, among others elements, with which it is possible to categorize different types of astronomical objects.

Although spectroscopy is a powerful technique for the study of astronomical objects, limitations in terms of the exposure time needed to obtain an acceptable signal-to-noise ratio in each wavelength band makes it unfeasible to use this technique on all celestial objects that are seen every night in the sky.

2.1.2 Photometry

Unlike spectroscopy, which decomposes light into its spectral components, photometry does not allow detailed detection of absorption or emission lines and uses broad filters to measure the total amount of light emitted by an astronomical source in a given wavelength range. The information extracted from photometry gives the possibility to produce light curves, which represent the variability in light intensity with respect to time. In particular, this process is performed using different specific ranges of light frequencies (photometric bands) by means of optical filters located in the telescopes, which provides a more complete profile to characterize a celestial object. Because information is available for the same object in different bands or filters, a light curve in different bands can be viewed as a multivariate time series with irregular observations for each photometric band.

2.1.3 Photometric Flux

The measurement obtained through the photometer corresponds to the photometric flux, which is a measure of the amount of emitted light by a source in a given unit of time and area. This measurement can be obtained by counting the amount of photons passing through an aperture and falling on a detector. In general, there are several factors that can cause errors in brightness measurements in astronomy, some of these are: uncertainty about the mean sky level due to noise in the annular space and Poisson noise due to the Poissonian nature of photon emission (Larsen, 2010; McCracken, 2017). Therefore, it is common to associate the photometric flux measurement , denoted by $F_s \geq 0$, with an error estimate, represented by $\sigma \geq 0$. Careful calibration of observing instruments and repeated measurements of a celestial object help to minimize these errors.

2.1.4 Apparent Magnitude

Photometric flux measurement can be used to calculate the apparent magnitude of a celestial object. Magnitude is represented through a logarithmic scale in which an increase of 5 units represents a 100-fold decrease in the amount of light emitted by the object (Chromey, 2016).

To calculate the apparent magnitude, the measurements of the photometric flux of the celestial object are compared with the measurements of a known reference star. The apparent magnitude can be calculated using the following formula:

$$m = -2.5 \; log_{10}(\frac{F_s}{F_{ref}}) + m_{ref}, \tag{2.1}$$

where m is the apparent magnitude of the celestial object, F_s is its measured photometric flux, F_{ref} is the measured photometric flux of the reference star, and m_{ref} is the known apparent magnitude of the reference star. Like flux measurement, the apparent magnitude has an associated observation error. Therefore, magnitude measurements are expressed in the form $m \pm \sigma_m$, where $\sigma_m \geq 0$ is the magnitude of the observational error.

2.1.5 Variable Stars

Variable stars are stars that change their brightness or apparent magnitude regularly or irregularly with time. These changes in the brightness may be due to a variety of factors, including variability in the rate of energy production in the star's core, variability in the amount of material in the star's surface, or the presence of nearby stars (Percy, 2007).

There are many types of variable stars, but some of the most common are as follows:

- **Cepheids:** are regular variable stars with periods of variation in brightness that can be used to measure the distances to galaxies and star clusters.
- **Eruptive flares:** are variable stars that undergo repetitive flares that suddenly and temporarily increase in brightness.
- **Eclipsants:** are variable stars where the variability in brightness is due to the occlusion of a nearby star.
- **RR Lyrae:** are regular variable stars that are common in ancient galaxies and are used as indicators of the distance to them.
- **Miras:** are variable stars of spectral type M that undergo slow and regular changes in brightness over months or years.
- **Novae:** are variable stars that undergo repetitive and explosive flares that suddenly and temporarily increase in brightness. Unlike Eruptive Flares, Novae has flares much more intense and extensive eruptions.
- **Eclipsing binaries:** are variable stars consisting of two stars in elliptical orbit that undergo changes in brightness due to mutual occlusion.

The classification of astronomical objects holds significant importance for multiple reasons: (i) it contributes to an enhanced comprehension of stellar physics, (ii) it aids in measuring distances to remote galaxies through the comparison of apparent magnitude with intrinsic magnitude, (iii) it enables astronomers to discern patterns and trends in the variability of stars, and (iv) it facilitates the study of the distribution and abundance of diverse categories of variable stars in distinct regions of the universe, thereby providing valuable insights into the formation and evolution of galaxies as well as stellar evolution.

2.2 Deep Learning

Deep learning is a subarea of machine learning based on artificial neural networks (ANN) with hidden layers. The particularity of these networks is that they can extract complex patterns from large amounts of data solving tasks such as: image recognition, natural language processing (NLP), text generation and audio classification, among others.

2.2.1 Perceptron

The fundamental building unit of an ANN is the perceptron, which is an artificial neuron that seeks to mathematically model the behavior of a human neuron. This concept was originally developed by Rosenblatt (1957). The perceptron is defined as follows:

$$\hat{y} = \sigma(\boldsymbol{w}^T \boldsymbol{x} + b), \tag{2.2}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ represents the input vector containing n values, $\boldsymbol{w} \in \mathbb{R}^n$ represents the parameters of the perceptron, b the intercept or bias, and $\hat{y}$ represents the predicted output of the model. The function σ, known as the activation function, is a nonlinear function that transforms the result of the linear transformation and determines the output of the perceptron.

2.2.2 Multilayer Perceptron (MLP)

The combination of several perceptrons, with interconnected hidden layers, forms a multilayer perceptron (MLP, Rumelhart et al. 1986) and is a specific type of Feed-Forward Neural Network (FNN), where the hidden layer units perform nonlinear transformations on the input data. Mathematically, the MLP can be defined as:

$$\hat{\boldsymbol{y}} = \sigma(\boldsymbol{W}^T \boldsymbol{x} + \boldsymbol{b}), \tag{2.3}$$

where $\boldsymbol{W} \in \mathbb{R}^{m \times n}$ represents the matrix of parameters called weights, $\boldsymbol{x} \in \mathbb{R}^n$ is the input of the layer, $\hat{\boldsymbol{y}} \in \mathbb{R}^m$ is the output of a layer of m neurons, $\boldsymbol{b} \in \mathbb{R}^m$ is a vector containing the bias of each neuron and σ is the activation function.

There are a wide variety of activation functions used in deep learning, however, the most commonly used are the sigmoidal (Sigmoid), hyperbolic tangent (Tanh) and rectifying linear unit (ReLU) functions (Dubey et al., 2022). The equations are shown below:

$$\text{Sigmoid}\left(\boldsymbol{x}\right) = \frac{1}{1+e^{-\boldsymbol{x}}}, \tag{2.4}$$

$$\text{Tanh}\left(\boldsymbol{x}\right) = \frac{e^{\boldsymbol{x}} - e^{-\boldsymbol{x}}}{e^{\boldsymbol{x}} + e^{-\boldsymbol{x}}}, \tag{2.5}$$

$$\text{ReLU}\left(\boldsymbol{x}\right) = \max\{0, \boldsymbol{x}\}. \tag{2.6}$$

Training a multilayer perceptron involves updating the weights associated with each connection among neurons to minimize a loss function that measures the difference between predicted and real outputs. This process is performed using different optimization algorithms, however, one of most common is gradient descent (Sun et al., 2019). In this framework, the most frequent loss function for regression and classification are the mean square error and the categorical cross-entropy loss function respectively (Wang et al., 2020), which are defined as:

$$MSE\left(w\right) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}_i\right)^2, \tag{2.7}$$

$$CCE(w) = -\sum_{i=1}^{N}\sum_{c=1}^{C} y_{ic}\log(\hat{y}_{ic}), \tag{2.8}$$

where $\hat{y}_i$ corresponds to the model output for data i, y_i corresponds to the actual desired output for data i, C is the number of classes, and N is the total number of data used to calculate the loss function.

2.2.3 Recurrent Neural Network (RNN)

When using sequential or time series data, MLPs cannot be used directly for learning and prediction. Recurrent Neural Networks (RNN; Rumelhart et al. 1986) are a variant of MLPs that can handle and predict patterns in data that have a temporal order. In particular, these networks add the concept of “memory”, which helps them store the states or information of previous inputs to generate the next output of the sequence.

Vanilla RNN

Vanilla RNN is the most basic type of recurrent neural network, in which the input from one layer serves as the output for the next. In simple terms, the RNN has two inputs: the current input and the hidden state of the previous layer. Mathematically, it can be expressed as (note that $\boldsymbol{W}$ represents independent weight matrices):

$$\boldsymbol{h}_t = f_h(\boldsymbol{x}_t, \boldsymbol{h}_{t-1}) = \sigma_h(\boldsymbol{W}_{ih}^T\boldsymbol{x}_t + \boldsymbol{b}_i + \boldsymbol{W}_{hh}^T\boldsymbol{h}_{t-1} + \boldsymbol{b}_h), \tag{2.9}$$

$$\hat{\boldsymbol{y}}_t = f_o(\boldsymbol{h}_t) = \sigma_o(\boldsymbol{W}_{hy}^T\boldsymbol{h}_t + \boldsymbol{b}_y), \tag{2.10}$$

where the equation (2.9) computes current state vector $\boldsymbol{h}_t \in \mathbb{R}^{d_h}$ as a function of the current input vector $\boldsymbol{x}_t \in \mathbb{R}^{d_x}$ and the hidden state vector of the previous layer $\boldsymbol{h}_{t-1}$. In this framework, d_x and d_h represent the dimension of the input and state vector respectively. The computation is performed using the weights matrices $\boldsymbol{W}_{ih}$ and $\boldsymbol{W}_{hh}$, and the bias vectors of $\boldsymbol{b}_i$ and $\boldsymbol{b}_h$. Also, an arbitrary nonlinear activation function σ_h is used to induce nonlinear interactions. Finally, the observed output of the system $\hat{\boldsymbol{y}}_t$ is modeled by a σ_o function that uses as input the current state $\boldsymbol{h}_t$ together with the weight matrix $\boldsymbol{W}_{hy}$ and the bias $\boldsymbol{b}_y$.

A vanilla RNN is suitable for problems with short and static sequences, but has difficulty retaining long-term information in longer sequences due to gradient backpropagation (Hochreiter, 1998). This means that important features in the initial input are lost as more cells are processed.

Long Short-Term Memory (LSTM)

Long Short-Term Memory (Hochreiter and Schmidhuber, 1997) cells, unlike RNNs, introduce “gates” that control the flow of information allowing better control and preservation of information over time. LSTM cell can be described as (note that $\boldsymbol{W}$

represents independent weight matrices):

$$\boldsymbol{i}_t = \sigma(\boldsymbol{W}_{ii}^T\boldsymbol{x}_t + \boldsymbol{b}_{ii} + \boldsymbol{W}_{hi}^T\boldsymbol{h}_{t-1} + \boldsymbol{b}_{hi}), \tag{2.11}$$

$$\boldsymbol{f}_t = \sigma(\boldsymbol{W}_{if}^T\boldsymbol{x}_t + \boldsymbol{b}_{if} + \boldsymbol{W}_{hf}^T\boldsymbol{h}_{t-1} + \boldsymbol{b}_{hf}), \tag{2.12}$$

$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}_{io}^T\boldsymbol{x}_t + \boldsymbol{b}_{io} + \boldsymbol{W}_{ho}^T\boldsymbol{h}_{t-1} + \boldsymbol{b}_{ho}), \tag{2.13}$$

$$\boldsymbol{g}_t = \tanh(\boldsymbol{W}_{ig}^T\boldsymbol{x}_t + \boldsymbol{b}_{ig} + \boldsymbol{W}_{hg}^T\boldsymbol{h}_{t-1} + \boldsymbol{b}_{hg}), \tag{2.14}$$

$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \boldsymbol{g}_t, \tag{2.15}$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t), \tag{2.16}$$

where $[\boldsymbol{W}_{ii}, \boldsymbol{W}_{hi}, \boldsymbol{b}_{ii}, \boldsymbol{b}_{hi}]$, $[\boldsymbol{W}_{if}, \boldsymbol{W}_{hf}, \boldsymbol{b}_{if}, \boldsymbol{b}_{hf}]$, and $[\boldsymbol{W}_{io}, \boldsymbol{W}_{ho}, \boldsymbol{b}_{io}, \boldsymbol{b}_{ho}]$ are the weights and biases of the input, forget and output gates, respectively. Likewise, $\boldsymbol{g}_t$ is the vector of candidates that could be added to the new cell state $\boldsymbol{c}_t$, where $\odot$ is the Hadamard product (Davis, 1962). Finally, the selection of elements $\boldsymbol{c}_t$ to be stored in the current state $\boldsymbol{h}_t$ is performed in Equation (2.16).

2.2.4 Transformers

The transformer model was first proposed in Vaswani et al. (2017) to solve the problems of previous architectures in NLP. Theses problems corresponds to the variable length sequences, long-term information retention and parallelization of temporal information processing (Bahdanau et al., 2014; Cho et al., 2014a,b; Sutskever et al., 2014). For this purpose, the transfomers are composed of two main modules: self-attention and positional encoding.

Multi-Head Self-Attention

The objective of the self-attention module is to improve the model's ability to handle variable length input sequences and capture long-term relationships in the input. This component is formulated using three different representations: query (Q), key (K) and value (V). Given an element Q and a list of K-V pairs, self-attention estimates the relevance of an element with respect to the others by computing a similarity between the Q and each of the K. This similarity is used to take a weighted average of all V, which allows relating different positions of the same sequence to compute a representation of the sequence. This three representations are input transformations such that $Q = \boldsymbol{E}(\boldsymbol{x})_i \boldsymbol{W}^Q$, $K = \boldsymbol{E}(\boldsymbol{x})_i \boldsymbol{W}^K$, and $V = \boldsymbol{E}(\boldsymbol{x})_i \boldsymbol{W}^V$, where $\boldsymbol{E}(\boldsymbol{x})$ represents the input token embedded within a high-dimensional space and $\boldsymbol{W}^Q$, $\boldsymbol{W}^K$, and $\boldsymbol{W}^V \in \mathbb{R}^{d_x \times d_k}$ are trainable weights matrices. The equation that describes this mechanism can be expressed as:

$$e_{ij} = \frac{\boldsymbol{E}(\boldsymbol{x})_i \boldsymbol{W}^Q (\boldsymbol{E}(\boldsymbol{x})_j \boldsymbol{W}^K)^T}{\sqrt{d_k}}, \tag{2.17}$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{l=1}^{L} \exp(e_{il})}, \tag{2.18}$$

$$\boldsymbol{z}_i = \sum_{j=1}^{L} \alpha_{ij} (\boldsymbol{E}(\boldsymbol{x})_j \boldsymbol{W}^V), \tag{2.19}$$

where d_k is a hyperparameter that specifies the embedding size of the self-attention head. The term e_{ij} is the scaled dot-product attention score between the i-th and j-th elements of the sequence, α_{ij} are the attention weights, representing how much attention the model should pay to the j-th element when processing the i-th element, and $\boldsymbol{z}_i$ represents the output vector for the i-th element from an h attention head, where h is a member of the set $\{1, \ldots, H\}$, indicative of the available attention heads.

Multi-head attention is generally employed, as it enables the model to simultaneously process information from distinct representation subspaces at different positions. The output from each head is concatenated (denoted as "Concat") and subsequently projected, facilitating the integration of information across various heads. This process is mathematically represented as:

$$\boldsymbol{z}_i = \mathrm{Concat}\left(\boldsymbol{z}_i^{(1)}, ..., \boldsymbol{z}_i^{(H)}\right) \boldsymbol{W}^O, \tag{2.20}$$

where $\boldsymbol{W}^O \in \mathbb{R}^{H \cdot d_k \times d}$ is the matrix of trainable weights.

As can be seen, the self-attention module does not incorporate sequence order, i.e., it is permutation invariant. However, preserving temporal order is important for any task that contains a sequence of items. For example, natural language is well structured and word order is important for language comprehension (Sutskever et al., 2014). The following Subsection shows the most common way to incorporate positional information into a transformer model.

Positional Encoding

Unlike RNNs that process data from a sequence in a recurrent manner, the positional encoding (PE; Vaswani et al., 2017) is a simple technique that allows exploiting the parallel computation of self-attention layers and aims to encode and preserve temporal information within an ordered sequence of data. The original transformer paper considered absolute positional encodings, which can be defined as:

$$\boldsymbol{PE}_{(pos,\ 2i)} = \sin(\boldsymbol{\omega}_{2i} \cdot pos), \tag{2.21}$$

$$\boldsymbol{PE}_{(pos,\ 2i+1)} = \cos(\boldsymbol{\omega}_{2i} \cdot pos), \tag{2.22}$$

$$\boldsymbol{\omega}_i = \frac{2\pi}{T^{\frac{2i}{d_x}}}, \tag{2.23}$$

where $\boldsymbol{\omega}_i$ is the angular frequency denoted in Equation (2.23), T is the period of the function, pos is the position, $i \in [0, ..., \frac{d_x}{2}]$ is the dimensionality of the PE, and each position is represented by the matrix $\boldsymbol{PE} \in \mathbb{R}^{L \times d_x}$. Thus, this module vectorially reproduces each position of a sequence in sine and cosine functions keeping the values in a normalized range between $[-1, 1]$. In particular, each position is represented at different angular frequencies with wavelengths from 2π to $2\pi \times T$, which allows to generate a unique position representation for each word of a sequence.

Input Representation

Usually, there are two categories of positional encodings that represent positional information in transformers, absolute and relative positional encoding. Both change the way the attention blocks receive the input representation (contextual and positional information).

The absolute positional encoding was proposed in Vaswani et al. (2017), where an embedded word with the positional encoding is directly added to generate the input representation $\boldsymbol{S}_i$:

$$\boldsymbol{S}_i = \boldsymbol{E}(\boldsymbol{x})_i + \boldsymbol{PE}_i, \tag{2.24}$$

$$e_{ij} = \frac{\boldsymbol{S}_i \boldsymbol{W}^Q (\boldsymbol{S}_j \boldsymbol{W}^K)^T}{\sqrt{d_k}}, \tag{2.25}$$

$$\boldsymbol{z}_i = \sum_{j=1}^{L} \alpha_{ij} (\boldsymbol{S}_j \boldsymbol{W}^V), \tag{2.26}$$

where $\boldsymbol{S} \in \mathbb{R}^{L \times d_x}$ is the sum of the contextual information passed through an embedding layer and the positional information passed through the positional encoding. The other terms were described in the Subsections above, in equation (2.17) and (2.19).

The relative positional encoding was proposed in Shaw et al. (2018), where they incorporate representations of relative positions directly into the self-attention mechanism of the transformer to encode the distance between any two positions. The formulation is as follows:

$$e_{ij} = \frac{\boldsymbol{E}(\boldsymbol{x})_i \boldsymbol{W}^Q (\boldsymbol{E}(\boldsymbol{x})_j \boldsymbol{W}^K + \boldsymbol{a}_{ij}^K)^T}{\sqrt{d_k}}, \tag{2.27}$$

$$\boldsymbol{z}_i = \sum_{j=1}^{L} \alpha_{ij} (\boldsymbol{E}(\boldsymbol{x})_j \boldsymbol{W}^V + \boldsymbol{a}_{ij}^V), \tag{2.28}$$

$$\boldsymbol{a}_{ij}^K = \boldsymbol{w}_{\text{clip}(j-i,r)}^K, \tag{2.29}$$

$$\boldsymbol{a}_{ij}^V = \boldsymbol{w}_{\text{clip}(j-i,r)}^V, \tag{2.30}$$

$$\text{clip}(j-i, r) = \max(-r, \min(r, j-i)), \tag{2.31}$$

where $\boldsymbol{a}_{ij}^K$, $\boldsymbol{a}_{ij}^V \in \mathbb{R}^{d_x}$ models the interaction between positions i and j. In particular, relative positions within a clipping distance r are learned. Thus, $\boldsymbol{w}_{\text{clip}(j-i,r)}^K$, $\boldsymbol{w}_{\text{clip}(j-i,r)}^V \in \mathbb{R}^{d_x}$ for $-r \leq \text{clip}(j-i, r) \leq r$ for a maximum relative distance r. This method can be viewed as a

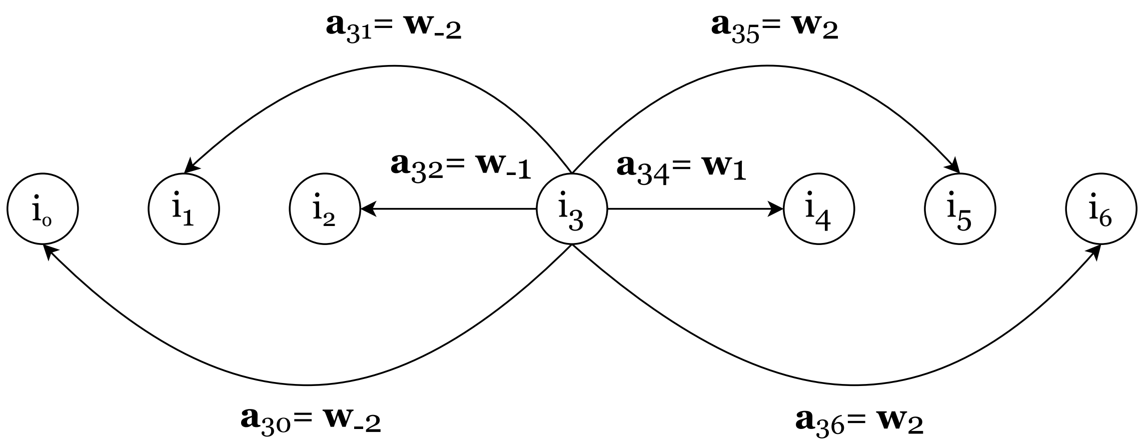

Figure 2.1: Example of interactions between various positions relative to a specific position.

labeled, directed, fully-connected graph, where the edge between the input elements i and j will be represented by the vectors $\boldsymbol{a}_{ij}^{K}$ and $\boldsymbol{a}_{ij}^{V}$. Figure 2.1 illustrates an example for position 3, taking into account only 7 observations and a maximum r of 2. As depicted, each pair of positions $i - j$ is assigned an embedding based on their relative difference. Similarly, for positions greater than r or less than $-r$, the same weight vector $\boldsymbol{w}r$ or $\boldsymbol{w}{-r}$ is applied, respectively. This approach limits the number of parameters and aids in generalizing across different sequence lengths. Specifically, it enables the model to learn distinct interactions based on the relative distances between tokens.

In general, the performance of both categories depends on the task to be solved. For example, Wang et al. (2021) conclude that absolute position embeddings are favorable for classification tasks and relative embeddings perform better for span prediction tasks.

Masked Language Model

Transformers are typically trained on large amounts of data to learn contextual representations using a self-supervised objective depending on the task to be solved, such as Causal Language Modeling (CLM), Masked Language Modeling (MLM), and Gap Sentence Generation (GSG), among others. In particular, we use MLM. The goal of this self-supervised objective is to train a language model to understand and predict the context of words in a sentence. For this purpose, a certain percentage (typically 15%) of the tokens in a text sequence are randomly selected and replaced with a special [MASK] token. This process creates a “corrupted” version of the original sentence, where some words are hidden. The task of the model is to predict the original tokens that have been masked out, based only on the context provided by the unmasked tokens. This requires the model to develop a deep understanding of language and context. During training, the model’s objective is to maximize the probability of correctly predicting the masked tokens. Mathematically, it can be defined as follows: Given a sequence $\boldsymbol{X} = \{x_i\}_{i=1}^{n}$, the input is corrupted by hiding a certain percentage of random tokens to obtain $\hat{\boldsymbol{X}} \subset \boldsymbol{X}$. Then, a transformer model parameterized by θ is trained and $\boldsymbol{X}$ is reconstructed by predicting the masked tokens $\hat{x}$ conditioned on $\hat{\boldsymbol{X}}$:

$$\max_{\theta} \log p_{\theta}(\boldsymbol{X}|\hat{\boldsymbol{X}}) = \max_{\theta} \sum_{i \in M} \log p_{\theta}(\hat{x}_i = x_i|\hat{\boldsymbol{X}}), \tag{2.32}$$

where M represents the set of indices of the masked tokens within the sequence, and p_{θ} denotes the probability of the sequence $\boldsymbol{X}$ given the corrupted sequence $\hat{\boldsymbol{X}}$, as modeled by the transformer with parameters θ.

Chapter 3

Methodology

This chapter describes the baseline astronomical transformer employed in this thesis, the various positional encodings subjected to testing with this architecture, the specifics of the training procedure, and comprehensive description of the data used.

3.1 Baseline Light Curve Transformer Model

Consider light curves of L observations, defined by a vector of magnitudes $\boldsymbol{x} \in \mathbb{R}^L$ and times $\boldsymbol{t} \in \mathbb{R}^L$ (MJD; Modified Julian Date). A standard transformer add up the projections of the observational and temporal data into a single vector:

$$\boldsymbol{S} = \boldsymbol{FFN}(\boldsymbol{x}) + \boldsymbol{PE}(\boldsymbol{t}), \tag{3.1}$$

where $\boldsymbol{FFN}(\boldsymbol{x}) \in \mathbb{R}^{L \times d_x}$ represents a FFN[1], d_x is the output size of this network, and $\boldsymbol{PE}(\boldsymbol{t}) \in \mathbb{R}^{L \times d_{pe}}$ is the temporal information passed through a positional encoding. To perform the addition operation, d_{pe} must equal to d_x. As proposed by Donoso-Oliva et al.

[1]Following Donoso-Oliva et al. (2022), we use no activation function for $\boldsymbol{FNN}(\boldsymbol{x})$.

(2022), the original positional encoding is expressed as a non-trainable function:

$$\boldsymbol{PE}(\boldsymbol{t})\ _{2j} = \sin\ (\boldsymbol{\omega}_{2j} \cdot \boldsymbol{t}), \tag{3.2}$$

$$\boldsymbol{PE}(\boldsymbol{t})\ _{2j+1} = \cos\ (\boldsymbol{\omega}_{2j+1} \cdot \boldsymbol{t}), \tag{3.3}$$

$$\boldsymbol{\omega}_j = \frac{2\pi}{1000^{\frac{j}{d_{pe}}}}, \tag{3.4}$$

where $\boldsymbol{\omega}_j$ are the angular frequencies, 1000 defines the lower bound of frequencies, $\boldsymbol{t}$ is the times vector, and $j \in [0, ..., d_{pe} - 1]$ are the dimensions of PE with a maximum of d_{pe} frequencies. This PE is a slight modification of the one proposed by Vaswani et al. (2017).

The self-attention blocks receive the matrix resulting from the previous step and can be expressed as:

$$e_{ij} = \frac{\boldsymbol{S}_i \boldsymbol{W}^Q \left(\boldsymbol{S}_j \boldsymbol{W}^K\right)^T}{\sqrt{d_k}}, \tag{3.5}$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{l=1}^{L} \exp(e_{il})}, \tag{3.6}$$

$$\boldsymbol{z}_i = \sum_{j=1}^{L} \alpha_{ij} \left(\boldsymbol{S}_j \boldsymbol{W}^V\right), \tag{3.7}$$

where $\boldsymbol{W}^Q$, $\boldsymbol{W}^K$, and $\boldsymbol{W}^V \in \mathbb{R}^{d_x \times d_k}$ are trainable weights matrices corresponding to the query (Q), key (K), and value (V), respectively, and d_k is a hyperparameter that specifies the embedding size of the self-attention head. The terms e_{ij}, α_{ij}, and $\boldsymbol{z}_i$ represent the similarity between the query and key vectors, the attention score and the output vector for each observation, respectively. The output of a self-attention block considers the information of the different heads h and is defined as:

$$\boldsymbol{z}_i = \text{Concat}\left(\boldsymbol{z}_i^{(1)}, ..., \boldsymbol{z}_i^{(H)}\right) \boldsymbol{W}^O, \tag{3.8}$$

where $\boldsymbol{W}^O \in \mathbb{R}^{H \cdot d_k \times d}$ is a trainable weight matrix, and d is the dimension of the output

$\boldsymbol{z} \in \mathbb{R}^{L \times d}$. This output can be used as input to other multi-head attention blocks to enable the model to capture dependencies at multiple levels of abstraction. The final representation is obtained in the last block. This representation is generated by a decoder FFN that reconstructs the input magnitudes $\hat{\boldsymbol{x}} \in \mathbb{R}^{L}$ in a self-supervised objective task, by minimizing the RMSE loss function. The resulting representation can serve as input for subsequent tasks, such as classification or regression.

3.2 Positional Encodings

This section presents all the positional encodings tested in ASTROMER and introduces a new approach to incorporate the temporal information.

3.2.1 Trainable

In order to capture temporal relationships from the light curves, we replaced the angular frequencies with an embedding layer consisting of d_{pe} trainable parameters. These parameters were initialized with the predefined frequencies given in Eq. (3.4).

3.2.2 Fourier

We enhance the learnability and flexibility of the positional representation in Eqs. (3.2) and (3.3) by modulating it with a MLP (Li et al., 2021):

$$\hat{\boldsymbol{PE}}(\boldsymbol{t}) = \left(\Phi_{\text{GeLU}}\left(\boldsymbol{PE}(\boldsymbol{t}) \cdot \boldsymbol{W}_1 + \boldsymbol{b}_1\right)\right) \cdot \boldsymbol{W}_2 + \boldsymbol{b}_2, \qquad (3.9)$$

where $\boldsymbol{W}_1 \in \mathbb{R}^{d_{pe} \times d_l}$ and $\boldsymbol{W}_2 \in \mathbb{R}^{d_l \times d_{pe}}$ are trainable parameters, $\boldsymbol{b}_1 \in \mathbb{R}^{d_l}$ and $\boldsymbol{b}_2 \in \mathbb{R}^{d_{pe}}$ are biases, and Φ_{GeLU} is the activation function. Here, d_l is the number of neurons in the hidden layer. In particular, $\boldsymbol{W}_2$ projects the representation to the dimension of the input embeddings.

3.2.3 Recurrent

We followed the approach of Nyborg et al. (2022) and used a Gated Recurrent Unit (GRU; Cho et al. 2014b) to incorporate temporal dependencies at time steps t_i expressed with the baseline positional encoding shown in Eqs. (3.2) and (3.3):

$$\boldsymbol{O(t)} = \mathrm{GRU}\left(\boldsymbol{PE(t)}\right), \tag{3.10}$$

$$\hat{\boldsymbol{PE}}\boldsymbol{(t)} = \boldsymbol{O(t)} \cdot \boldsymbol{W}_p + \boldsymbol{b}_p, \tag{3.11}$$

where $\boldsymbol{O(t)} \in \mathbb{R}^{L \times d_{pe}}$ is the output of the GRU at each time step, $\boldsymbol{W}_p \in \mathbb{R}^{d_{pe} \times d_{pe}}$ is a trainable weight matrix and $\boldsymbol{b}_p \in \mathbb{R}^{d_{pe}}$ is the trainable bias.

3.2.4 Tupe-A

We follow the approach employed in Ke et al. (2020) for NLP and separate the mixed correlations produced between observational and temporal information in the attention matrix by redefining Eqs. (3.5) and (3.7). To represent the queries and keys of the temporal information expressed by Eqs. (3.2) and (3.3), we introduce new parameters $U_q, U_k \in \mathbb{R}^{d_{pe} \times d_k}$, respectively:

$$e_{ij} = \frac{\boldsymbol{FFN(x)}_i \boldsymbol{W}^Q \left(\boldsymbol{FFN(x)}_j \boldsymbol{W}^K\right)^T}{\sqrt{d_k}} + \frac{\boldsymbol{PE(t)}_i \boldsymbol{U}^Q \left(\boldsymbol{PE(t)}_j \boldsymbol{U}^K\right)^T}{\sqrt{d_k}}, \tag{3.12}$$

$$\boldsymbol{z}_i = \sum_{j=1}^{L} \alpha_{ij} (\boldsymbol{FFN(x)}_j \boldsymbol{W}^V). \tag{3.13}$$

For efficiency, we share these new parameters across different multi-head attention blocks Ke et al. (2020).

3.2.5 Concat

Following the same idea and aiming to minimize the noise generated in the attention matrix due to mixed correlations between observational and temporal information, we concatenated them in separate orthogonal spaces:

$$\boldsymbol{S} = [\boldsymbol{FFN}(\boldsymbol{x}) \mid\mid \boldsymbol{PE}(\boldsymbol{t})]\,. \tag{3.14}$$

In this case, we utilize the trainable PE described in subsection 3.2.1.

3.2.6 PE on Attention (PEA)

To avoid mixing information, we propose incorporating positional encoding directly into the final representation obtained from the last multi-head attention block:

$$\hat{\boldsymbol{z}} = \boldsymbol{z} + \boldsymbol{PE}(\boldsymbol{t}), \tag{3.15}$$

where $\boldsymbol{PE}(\boldsymbol{t})$ is expressed by the baseline positional encoding shown in Eqs. (3.2) and (3.3) as a non-trainable function. Here, the multi-head attention block takes only the observational information $\boldsymbol{FFN}(\boldsymbol{x})$ to compute the attention.

3.3 Training Details

We divided this work into two experiments: the first involves analyzing various PEs tested in ASTROMER and their effects on pretraining stage and classification task; the second focuses on evaluating the adaptation of ASTROMER during the finetuning stage on a real dataset, which has a significantly different cadence, as described in subsection 3.4.

We ran the experiments on a Nvidia RTX A5000 GPU, employing two multi-head attention blocks with $H = 4$ heads and $d_k = 64$ neurons. The model dimensions were set at $d = d_x = d_{pe} = 256$ and $d_l = 64$, with the exception of Concat PE, which employed $d_x = d_{pe} = 128$. Light curve windows with a maximum length of $L = 200$ were considered. For light curves that exceeded this length, subsequent time windows were sampled, beginning from a random position. For light curves with fewer than L observations, zero values were padded at the end. Each generated window was subtracted from its observational and temporal mean, creating magnitude and time vectors with zero mean.

3.3.1 Experiment 1

For the pretraining and finetuning stages, we followed the strategy used in Devlin et al. (2018), masking a percentage of the observations in the light curves. Specifically, we selected 50% of the data for evaluating the reconstruction of the magnitude $\boldsymbol{x}$ in the loss function. Within this percentage, we masked 30% of the observations, replaced 10% with random values, and left the remaining 10% of observations visible. We used early stopping with patience of 40 epochs on the validation loss. The Adam optimizer (Kingma and Ba, 2014) was used with a learning rate of 10^{-5} and a batch size of $2,000$.

For the classification task, we used two hidden layers of 256 Long Short-Term Memory (LSTM; Hochreiter and Schmidhuber 1997) units followed by a MLP with a softmax activation function. The dimension of this output layer depends on the number of classes to be classified. We also divided the training and validation sets into 3 folds with an 80/20 ratio, respectively. We used early stopping with a patience of 20 epochs on the validation loss and the Adam optimizer with a learning rate of 10^{-4} and a batch size of 512.

3.3.2 Experiment 2

Since the relative relationships between magnitudes are dependent on astronomical phenomena, and the cadences are determined exclusively by the survey rather than the physical object being observed, we modify the common approach to finetuning. We experiment with different configurations when performing this stage. In particular, we tested finetuning only the positional encoding component and analyzed its effects compared to finetuning only the FNN corresponding to the magnitudes, as well as the full model.

For this stage, we used the same strategy used in Experiment 1. We selected 50% of the data for evaluating the reconstruction of the magnitude $\boldsymbol{x}$ in the loss function. Within this percentage, we masked 30% of the observations, replaced 10% with random values, and left the remaining 10% of observations visible. We used early stopping with patience of 40 epochs on the validation loss. The Adam optimizer was used with a learning rate of 10^{-5} and a batch size of $2,000$.

3.4 Data Description

The datasets employed in Experiment 1 for evaluating the pretraining stage and classification task are identical to the real datasets used in Donoso-Oliva et al. (2022). Conversely, the dataset used in Experiment 2, which aims to evaluate the adaptation of ASTROMER during the finetuning stage, is another real dataset characterized by a significantly shorter cadence compared to the one used for pretraining ASTROMER.

3.4.1 Experiment 1

For the pretraining stage, we utilized the MACHO light curves dataset (Alcock et al., 2000), a project aimed at searching for gravitational microlensing events and variable stars in the Milky Way. For this stage, we used unlabeled MACHO light curves and excluded those exhibiting noisy behavior[2]. The dataset comprised a total of 1,529,386 light curves in the R-band, with a median cadence of 1.00 days. The magnitude and cadence distributions can be seen in Figure 3.1. Subsequently, we evaluated the performance of the pretrained transformers on the classification task using a subset of 500 objects per class from the MACHO labeled survey (*Full* hereafter, Cutri et al., 2003). Figure A.1 in Appendix A shows various light curve samples from this dataset. To isolate the effect of cadence at this task, we simulated three datasets from the MACHO labeled subset by modifying the cadence of the light curves. Specifically, we removed observations from the light curves at

[2]We defined noise as points in the light curve with $|Kurtosis| > 10$, $|Skewness| > 1$, and a magnitude error > 0.1.

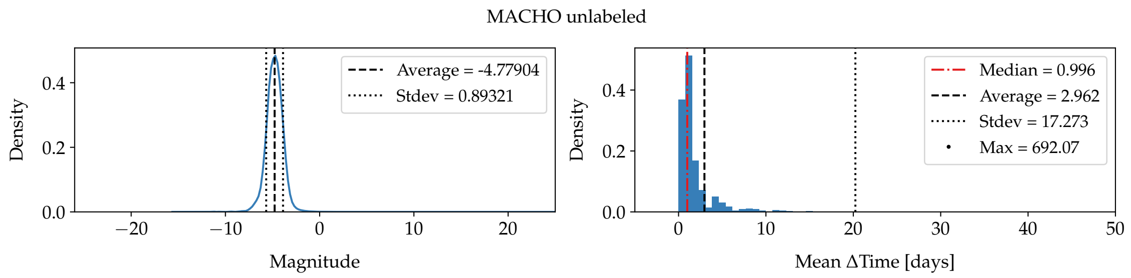

Figure 3.1: Unlabeled MACHO magnitude and cadence distributions.

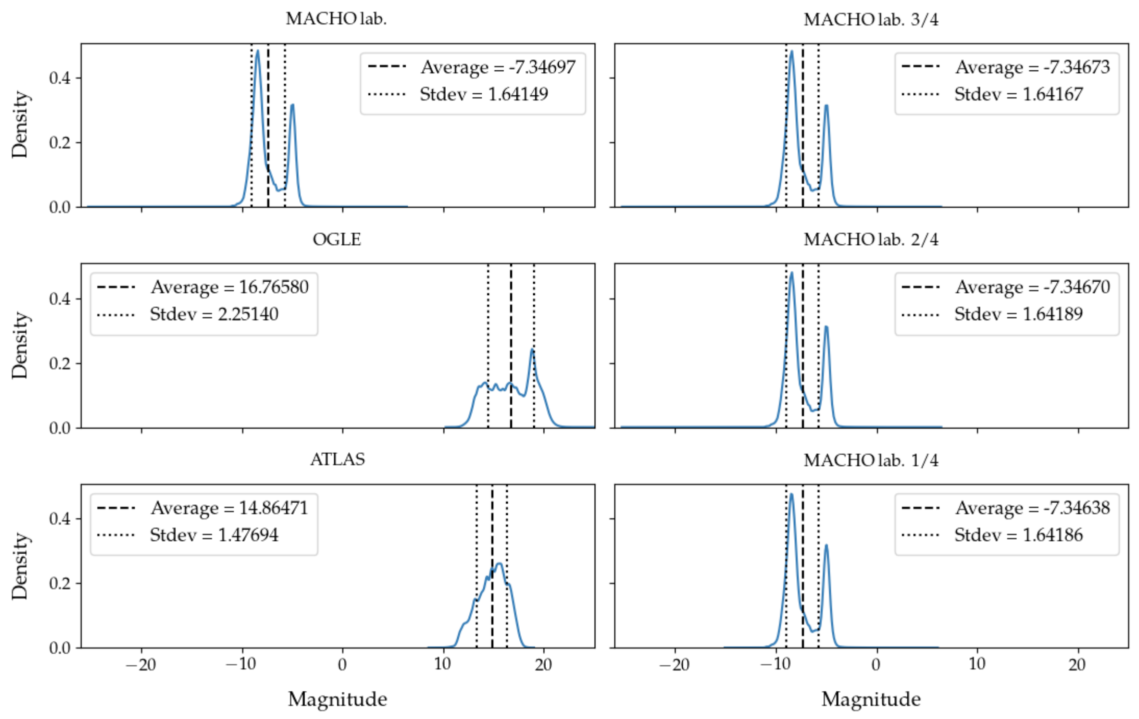

Figure 3.2: Magnitude distribution of the different data subsets.

rates of 3/4, 1/2, and 1/4, respectively. At a rate of 3/4, we removed the last observation out of four, while at a rate of 1/4, the last three observations were removed. An example of a light curve with observations removed at different rates can be seen in Figure A.2 in Appendix A.

To account for external factors, such as changes in the band distribution, we also tested the pretrained transformers on OGLE-III (Udalski, 2004), which was a variable star search project that focuses on the search for gravitational microlensing and eruptive flares and ATLAS (Heinze et al., 2018), which was an early warning system for terrestrial asteroid impacts that seeks to detect and characterize astronomical objects that may pose a risk to Earth. The OGLE-III data contains 358,288 I-band light curves and ATLAS data contains 422,630 orange-band light curves. Specifically, we used a subsample of 500 objects per class from each labeled data subsets to consider the scenario where we have few labeled data. Figures A.4 and A.3 in Appendix A illustrate different light curve samples from the OGLE and ATLAS datasets, respectively.

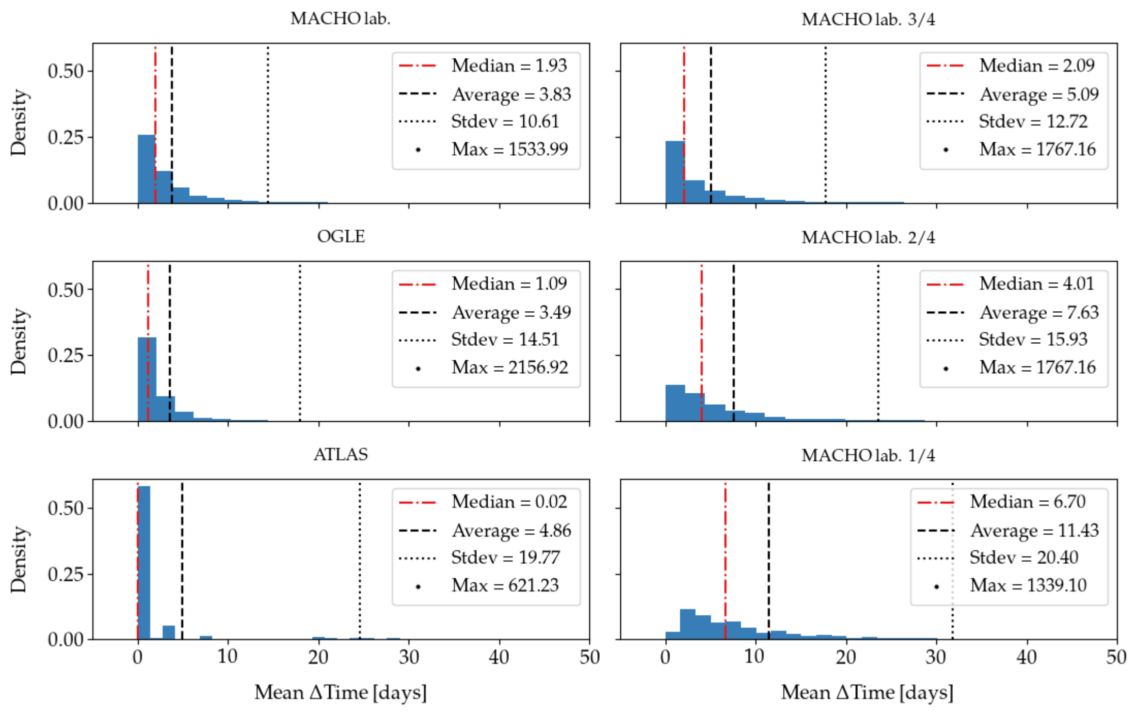

Figure 3.3: Cadence distribution of the different data subsets.

The magnitude distributions of the unlabeled MACHO data and the labeled datasets are presented in Figures 3.1 and 3.2, respectively. This comparison highlights the similarities in magnitude distributions across various labeled MACHO sets (full, 3/4, 1/2, and 1/4), and the differences in magnitude distributions between the OGLE, ATLAS, and labeled MACHO datasets. Similarly, the cadence distributions for the unlabeled and labeled data are shown in Figures 3.1 and 3.3, respectively. Key statistical measures are provided to understand the sampling frequency of observations within the light curves. The unlabeled MACHO dataset, which served as the pretraining data for the transformers, exhibits a median of 1.00 and a mean of 2.96 with a standard deviation of 17.19. These values indicate the time gap between successive observations and the temporal separation between groups of observations in the light curves. In particular, the unlabeled MACHO dataset demonstrates a smaller time gap compared to the labeled MACHO dataset, while also exhibiting a higher standard deviation, implying greater temporal separation between groups of observations. The labeled MACHO-derived datasets (3/4, 1/2, and 1/4) exhibit an increase in both the median and standard deviation as light curve observations are reduced. Additionally, they show distinctions compared to both the labeled and unlabeled MACHO datasets. Regarding OGLE, its cadence displays similarities with that of the unlabeled MACHO dataset. However, ATLAS exhibits more pronounced time gaps between groups of observations. The mean is influenced by the standard deviation, while the median indicates that the observations are taken at short time intervals.

Table 3.1 displays the classes used for each of the datasets. MACHO labeled has six classes, OGLE has ten classes, and ATLAS has four classes. The modified cadence datasets maintain the same number of classes as the MACHO labeled dataset.

Tag	MACHO lab.	OGLE	ATLAS
EC	Eclipsing Binary	Eclipsing Binary	-
ED	-	Detached Binary	Detached Binary
ESD	-	Semi-detached Binary	-
Mira	-	Mira	Mira
OSARG	-	Small-amplitude red giant	-
RRab	RR Lyrae type ab	RR Lyra type ab	Pulse
RRc	RR Lyrae type c	RR Lyrae type c	
dsct	-	Delta Scuti	
Cep_0	Cepheid type I	Cepheid	
Cep_1	Cepheid type II		
SRV	-	Semi-regular variable	-
LPV	Long period variable	-	-
CB	-	-	Close Binaries

Table 3.1: Labels from each of the datasets used in the classification task.

3.4.2 Experiment 2

To assess the adaptability of ASTROMER in a context where the cadence is markedly different from that of the dataset used for its pretraining, we employed data from “Kepler Mission II: Eclipsing Binaries in DR2” (Slawson et al., 2011), which aimed to discover Earth-like planets orbiting other stars. To focus exclusively on data with labels, we excluded the “unclassified” labels from this dataset, resulting in a total of 2,014 light curves. Figure A.5

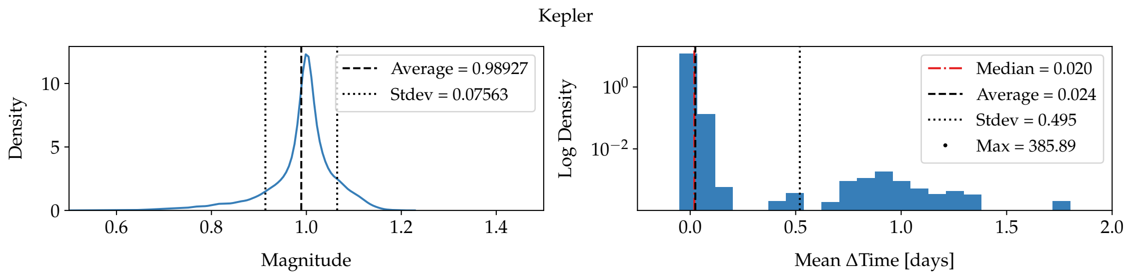

Figure 3.4: Kepler cadence distribution.

in Appendix A displays a light curve from each class in the Kepler dataset, along with the ID and cadence of each curve.

The magnitude and cadence distributions can be seen in Figure 3.4. It is noted that the magnitudes in this dataset range from 0.99 ± 0.076. These were obtained over a spectral range extending from the visible to the near-infrared, approximately from 430 nm to 890 nm. For the unlabeled MACHO dataset, the covered range was from about 550 nm to approximately 800 nm. Regarding the cadence, the general measurement rate was 0.020 days with a small standard deviation of 28.8 minutes. This shows a significant difference compared to the unlabeled MACHO dataset, which had observations taken every 1.00 day with a standard deviation of 17.273 days.

We also capitalize on the fact that this dataset contains observations every 30 minutes to generate an interpolation in time and match the cadence behavior of the real datasets (MACHO, OGLE, and ATLAS). This approach allows us to create three datasets with the same flow but with different and realistic cadences. Initially, we fit a kernel density estimate (KDE) to the cadence of each dataset, using a Gaussian kernel and cross-validation over the bandwidths. We find that the optimal bandwidths are $1,031$, $1,300$, and $0,100$ for MACHO, OGLE, and ATLAS, respectively. Subsequently, we take the first MJD of each light curve and sample different values from the KDE, accumulating these values until they match or fall below the last MJD of the light curves. Finally, we linearly interpolate the flux for each time point. Through this process, we obtained three sets of data with distinct cadences, which we named: 1) Kepler with MACHO cadence, 2) Kepler with OGLE cadence, and 3)

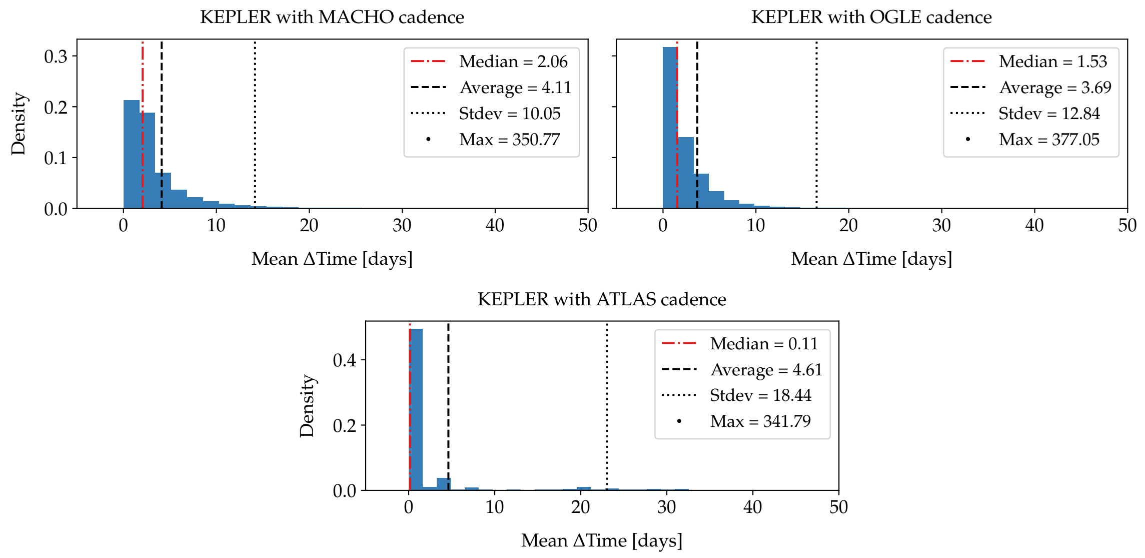

Figure 3.5: Cadence distribution of the different dataset modified from Kepler.

Kepler with ATLAS cadence. Figure 3.5 displays the cadence distribution of each of the generated datasets, which can be compared to the actual datasets in Figure 3.3.

Chapter 4

Results and Discussion

Table 4.1 provides the evaluation of a transformer pretrained from scratch using different positional encodings on both simulated and real datasets. We pretrained each transformer on the unlabeled MACHO data and evaluated the reconstruction of the observational information in terms of RMSE and training time. We then used the generated representation for training the classification layers on each labeled dataset and evaluated its performance in terms of F1-score. Our baseline is the fixed positional encoding described in subsection 3.1.

During pretraining, the Trainable PE demonstrated a slight improvement in terms of the reconstruction RMSE and a significant reduction in training time when compared to

PE type	MACHO unlab.		MACHO lab.				OGLE	ATLAS
			Full	3/4	1/2	1/4		
	RMSE	Time (epochs)	F1 (%)	F1 (%)	F1 (%)	F1 (%)	F1 (%)	F1 (%)
Baseline	.170	6d 14h (523)	71.6 ± 1.9	69.2 ± 1.9	66.2 ± 1.9	63.3 ± 1.5	71.3 ± 1.1	65.8 ± 1.4
Trainable	**.169**	2d 13h (202)	72.9 ± 2.1	72.3 ± 1.0	**71.0 ± 1.0**	**69.0 ± 0.5**	74.9 ± 1.4	65.4 ± 1.8
Fourier	.170	1d 20h (142)	73.0 ± 1.1	70.2 ± 1.9	67.8 ± 0.9	62.9 ± 2.0	72.0 ± 0.8	**69.6 ± 0.1**
Recurrent	.197	0d 16h (048)	67.1 ± 1.8	63.5 ± 2.5	59.7 ± 1.9	54.6 ± 1.3	70.7 ± 1.1	68.3 ± 0.9
Tupe-A	.219	0d 17h (084)	67.3 ± 1.6	66.1 ± 1.4	64.9 ± 1.0	60.8 ± 0.9	71.0 ± 1.0	67.5 ± 0.9
Concat	.170	3d 01h (237)	**73.4 ± 1.1**	**73.1 ± 1.7**	70.9 ± 1.7	**69.0 ± 1.8**	74.5 ± 1.3	68.1 ± 0.6
PEA	.199	0d 17h (058)	69.7 ± 0.9	68.9 ± 1.8	68.0 ± 1.0	65.5 ± 2.5	**76.3 ± 1.2**	66.9 ± 1.0

Table 4.1: Performance of different positional encodings in the pretraining stage and classification task.

the baseline. The Fourier and Concat PE matched the performance of the baseline while needing less computational resources. The Fourier PE showed the best performance in terms of both training time and reconstruction performance. Recurrent, Tupe-A, and PEA did not outperform the Baseline in RMSE terms, but converged in less than a day of training. Learning curves are shown in Figure B.1 in Appendix B.

Since we are analyzing which PE can generate a better representation during the pretraining stage, we keep the transformer, including the PE, fixed when training for the classification task. We start by evaluating the performance of the transformers on the MACHO labeled datasets considering the effect of the change in cadence. The Trainable PE outperformed the Baseline for all cadences. In particular, the degradation of results for sparser light curves is less severe with the Trainable PE than with the non-trainable one. Similarly, Fourier PE performs better than the baseline on three out of four datasets. However, the degradation of results, as evaluated on the 1/2 and 1/4 cadences, was similar to the baseline and worse than the Trainable PE. The Recurrent and Tupe-A PE show worse classification performance than the baseline. The Concat PE outperformed the Baseline and achieved three of the four best performances in terms of F1-score. Its degradation was minimal for sparser light curves, and close to the Trainable PE. Finally, PEA outperformed the baseline for cadences of 1/2 and 1/4, but did worse for the full and 3/4 cadences.

Upon adding changes in the magnitudes distributions using OGLE and ATLAS, we observe that the baseline exhibits inferior overall F1-score performance. The trainable PE model outperforms the baseline in OGLE and demonstrated a similar performance as the baseline for ATLAS. The Fourier PE model showed superior performance in both astronomical surveys with respect to the baseline. However, it did not outperform the Trainable PE model in OGLE. In particular, the Fourier PE model obtained the best F1-score in ATLAS. Similarly, the Recurrent and Tupe-A PE models outperformed the baseline model in ATLAS and while exhibiting a similar performance in OGLE. Finally, the Concat PE and PEA models outperformed the baseline in both OGLE and ATLAS, with the latter obtaining the highest F1-score among all the PE on OGLE.

The separation of temporal and observational information into orthogonal spaces (Concat PE) results in better classification performance on all datasets, yielding the best average F1-score overall. Recall that both the Trainable and Concat PE use the same trainable PE: the first add the PE to a vectorized representation of the magnitudes, while the second concatenates these vector. Both of these PEs show a small degradation in results for the MACHO datasets with different cadences, implying that they allow for a better representation of temporal information.

In terms of training time, all the trainable PEs and the proposed PEA exhibit reduced pretraining time compared to the baseline. Out of the three models that take less than one day to train (Recurrent, Tupe-A, and PEA), the best classification results for the different MACHO cadence datasets are achieved by our proposed PEA. At the same time, PEA outperforms the Recurrent and Tupe-A PEs when transfered to OGLE, while the three of them achieve similar classification results on the ATLAS dataset (less than 1.6 sigma). This is of particular importance when training large light curve models with massive datasets for next generation surveys such as the LSST.

Figure 4.1 demonstrates the baseline transformer's goodness of fit, pretrained on the MACHO dataset, for the magnitude reconstruction task on the Kepler dataset. Initially, we directly evaluate the pretrained transformer on the Kepler data, represented by the blue bar. Subsequent steps involve finetuning the pretrained transformer by training different

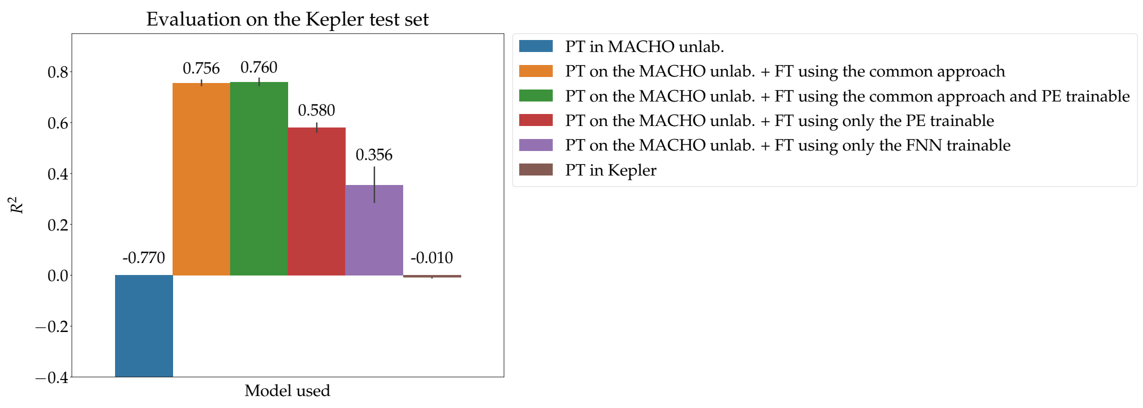

Figure 4.1: Evaluation of the goodness of fit for the various approaches tested.

model components. The orange bar indicates finetuning through the conventional approach of adjusting all model weights. The green bar illustrates finetuning that includes training the PE along with the common approach. The red bar depicts the finetuning process focused solely on adjusting the PE weights. Another purple bar represents the adjustment of weights for the FFN that projects the magnitudes. Lastly, the brown bar shows the baseline transformer being pretrained from scratch on the Kepler data.

Figure 4.1 reveals that using a model pretrained on the unlabeled MACHO dataset and directly evaluating it on the Kepler test data significantly degrades the results, leading to a poor fit of -0.725, attributable to drastic cadence changes. Finetuning using the common approach showed a significant improvement with a finetuning score of 0.756. Adding training of the trainable PE to the common approach resulted in an adjustment of 0.760, which is comparable to the scenario where this component is not considered. Training only the PE achieves a fit of 0.580, and training only the FFN achieves a fit of 0.356. This particularly indicates that the mismatch in the temporal component is more significant than in the observational aspect, which can also be attributed to the fact that magnitudes are influenced by astronomical phenomena, whereas cadences are determined by the survey rather than by the observed physical object. In contrast, training the transformer from scratch on the Kepler training data shows a better fit than using the pretrained model on the unlabeled MACHO dataset. However, the fit remains deficient and is not better than finetuning the pretrained model on a dataset with significantly different cadence. This deficiency is attributed to the limited size of Kepler's dataset, which consists of only 2,014 light curves, hindering the model's ability to extract sufficient information. This is particularly challenging given that transformer models require extensive data for pretraining relative to their parameter count.

We also evaluated the transformer, pretrained on the MACHO dataset, on Kepler data with cadences from MACHO, OGLE, and ATLAS, as depicted in Figure 4.2. The color coding of the bars follows the same legend as in Figure 4.1. The model was not pretrained from scratch on data with altered cadences, as this approach was not the focus of our work. Direct evaluation of the pretrained model on any dataset with changed cadence

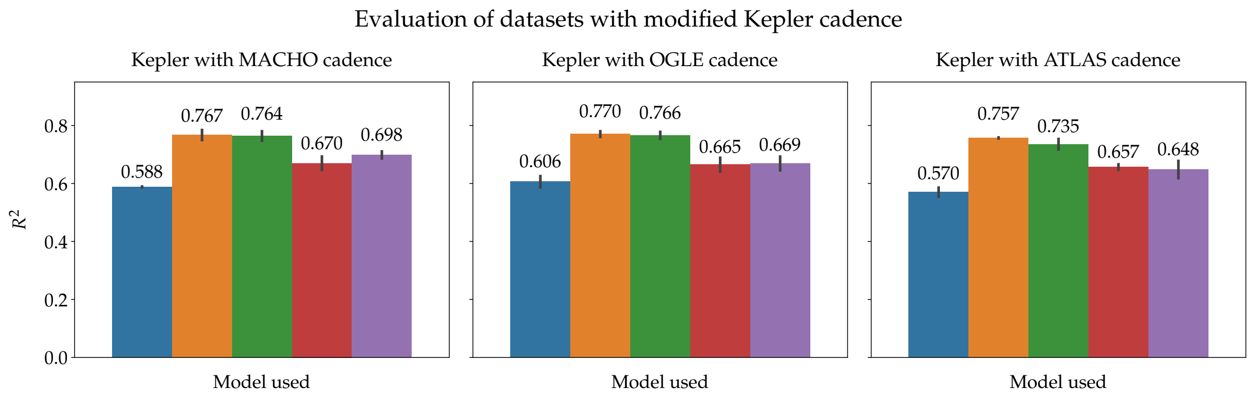

Figure 4.2: Evaluation of the goodness of fit for the modified Kepler cadence.

consistently showed a positive fit greater than 0.570. Employing the common approach, as well as augmenting it with training of the PE weights, enhanced the fit relative to the direct evaluation of the pretrained model, yielding very similar outcomes across all datasets. The best fit achieved was 0.770 in OGLE, with the lowest at 0.735 in ATLAS. Lastly, training solely the PE weights or the FFN weights resulted in improvements over the direct evaluation of the pretrained model. However, they did not surpass a fit of 0.700.

Using a dataset with a significantly different cadence from that on which the transformer was pretrained directly affected the model's fit. However, adjusting the transformer weights on the new dataset achieved results with an R^2 value of up to 0.770 within an average of 2,807 hours. This demonstrates the model's ability to adapt to a dataset with a markedly distinct cadence. Although finetuning the entire model yielded better results, this approach can be computationally demanding in terms of training time, GPU/CPU intensity, and additional RAM required for gradient storage during parameter updates. In contrast, training only the PE weights reduces computational demands (with only 256 parameters) and shortens the training time by an average of 0,948 hour. However, this method resulted in a slight decrease in performance compared to the common finetuning method. Simply changing the cadence improves direct evaluation but does not significantly enhance the fit when finetuning is performed on the dataset. Likewise, the gap between the effectiveness of adjusting PE weights and FFN weights is reduced, since the difference in cadence and magnitudes is not as pronounced as it was with the original Kepler dataset.

Chapter 5

Conclusion

This thesis has presented a comprehensive exploration into the adaptability and efficacy of various positional encodings (PEs) in transformer models for astronomical light curve analysis. Our empirical studies, centered on the light curve transformer model, provide significant insights into the model's performance across different astronomical surveys with varying cadences and magnitude distributions.

Our results have demonstrated that using trainable positional encoding offers advantages over a non-trainable PE baseline, both in terms of model performance and computational efficiency. The latter is particularly critical when dealing with vast astronomical datasets, such as those from the Vera C. Rubin Observatory. We have also highlighted the benefits of separating observational and temporal information within the attention matrix and proposed a new approach for incorporating temporal information directly into the output of the last attention layer. Additionally, we show that our proposed method trains faster than the baseline while achieving competitive classification performance. On the other hand, our experiments showed that a pretrained transformer can adapt to a dataset with a significantly different cadence and limited data. Similarly, adjusting the dataset to match the cadence for which the model was trained improves performance in the direct evaluation of the pretrained model. However, when performing finetuning, the goodness of fit is similar. Finally, training only the PE has emerged as a crucial strategy in adapting to datasets with markedly different cadences, such as the

Kepler dataset. This method not only reduced computational resources but also maintained respectable performance levels, achieving an R^2 of 0.580 in the reconstruction of light curves.

In light of our findings, it is evident that the choice of positional encoding in transformer models plays a pivotal role in enhancing the models' performance on astronomical time series data. Trainable positional encodings, particularly those that delineate temporal from observational information, present a promising avenue for future research in this domain. They strike a delicate balance between computational efficiency and model accuracy, a balance that is crucial for analyzing the vast volumes of data produced by contemporary astronomical surveys. For future work, exploring whether the embeddings generated by various finetuning techniques on the Kepler dataset and its derivatives are sufficiently representative to distinguish between classes in subsequent tasks could provide valuable insights. Additionally, assessing the adaptation of other positional encodings to datasets with markedly different cadences would extend our understanding of PE's versatility. A particularly intriguing direction would be the development and implementation of a positional encoding that can effectively capture the relative distances between irregular observations in a time series. Such an approach could significantly improve the model's ability to handle the intricacies of astronomical data.

As we stand on the brink of a new era in astronomical research, powered by advanced observatories and sophisticated data analysis techniques, this thesis contributes valuable insights into the development of more efficient and accurate models for light curve analysis. Our work lays the groundwork for further research in this area, potentially leading to groundbreaking discoveries in our understanding of the universe.

Bibliography

C. Alcock, R. Allsman, D. R. Alves, T. Axelrod, A. C. Becker, D. Bennett, K. H. Cook, N. Dalal, A. J. Drake, K. Freeman, et al. The macho project: microlensing results from 5.7 years of large magellanic cloud observations. *The Astrophysical Journal*, 542(1):281, 2000.

T. Allam Jr and J. D. McEwen. Paying attention to astronomical transients: Introducing the time-series transformer for photometric classification. 2022.

N. Astorga, I. Reyes, G. Cabrera, F. Förster, P. Huijse, J. Arredondo, D. Moreno-Cartagena, A. Muñoz-Arancibia, A. Bayo, M. Catelan, et al. Atat: Astronomical transformer for time series and tabular data. 2023.

D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

T. Charnock and A. Moss. Deep recurrent neural networks for supernovae classification. *The Astrophysical Journal Letters*, 837(2):L28, 2017.

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

K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014b.

F. R. Chromey. *To measure the sky: an introduction to observational astronomy.* Cambridge University Press, 2016.

R. e. Cutri, M. Skrutskie, S. Van Dyk, C. Beichman, J. Carpenter, T. Chester, L. Cambresy, T. Evans, et al. Vizier online data catalog. *II/246*, 3, 2003.

C. Davis. The norm of the schur product operation. *Numerische Mathematik*, 4(1):343–344, 1962.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

C. Donoso-Oliva, G. Cabrera-Vives, P. Protopapas, R. Carrasco-Davis, and P. A. Estévez. The effect of phased recurrent units in the classification of multiple catalogues of astronomical light curves. *Monthly Notices of the Royal Astronomical Society*, 505(4): 6069–6084, 2021.

C. Donoso-Oliva, I. Becker, P. Protopapas, G. Cabrera-Vives, H. Vardhan, et al. Astromer: A transformer-based embedding for the representation of light curves. *arXiv preprint arXiv:2205.01677*, 2022.

S. R. Dubey, S. K. Singh, and B. B. Chaudhuri. Activation functions in deep learning: A comprehensive survey and benchmark. *Neurocomputing*, 2022.

A. Heinze, J. L. Tonry, L. Denneau, H. Flewelling, B. Stalder, A. Rest, K. W. Smith, S. J. Smartt, and H. Weiland. A first catalog of variable stars measured by the asteroid terrestrial-impact last alert system (atlas). *The Astronomical Journal*, 156(5):241, 2018.

S. Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116, 1998.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

W. Huggins and W. A. Miller. On the spectra of some of the fixed stars. *Proceedings of the Royal Society of London*, pages 242–244, 1863.

Ž. Ivezić, S. M. Kahn, J. A. Tyson, B. Abel, E. Acosta, R. Allsman, D. Alonso, Y. AlSayyad, S. F. Anderson, J. Andrew, et al. Lsst: from science drivers to reference design and anticipated data products. *The Astrophysical Journal*, 873(2):111, 2019.

G. Ke, D. He, and T.-Y. Liu. Rethinking positional encoding in language pre-training. *arXiv preprint arXiv:2006.15595*, 2020.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

G. R. Kirchhoff and R. W. Bunsen. Chemische analyse durch spectralbeobachtungen. 2008.

S. S. Larsen. Astronomical magnitudes and photometry, May 2010. URL `https://www.astro.ru.nl/~slarsen/teaching/OA1UU/OA1_0910/Viewgraphs/pdf/lecture3.pdf`.

Y. Li, S. Si, G. Li, C.-J. Hsieh, and S. Bengio. Learnable fourier features for multi-dimensional spatial positional encoding. *Advances in Neural Information Processing Systems*, 34:15816–15829, 2021.

H. J. McCracken. An introduction to photometry and photometric measurements, January 2017. URL `http://www2.iap.fr/users/hjmcc/hjmcc-photom-ohp-2017.key.pdf`.

M. Morvan, N. Nikolaou, K. H. Yip, and I. Waldmann. Don't pay attention to the noise: Learning self-supervised representations of light curves with a denoising time series transformer. *arXiv preprint arXiv:2207.02777*, 2022.

D. Muthukrishna, G. Narayan, K. S. Mandel, R. Biswas, and R. Hložek. Rapid: early classification of explosive transients using deep learning. *Publications of the Astronomical Society of the Pacific*, 131(1005):118002, 2019.

J. Nyborg, C. Pelletier, and I. Assent. Generalized classification of satellite image time series with thermal positional encoding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1392–1402, 2022.

J. Pan, Y.-S. Ting, and J. Yu. Astroconformer: Inferring surface gravity of stars from stellar light curves with transformer. *arXiv preprint arXiv:2207.02787*, 2022.

J. Pasquet, J. Pasquet, M. Chaumont, and D. Fouchez. Pelican: deep architecture for the light curve analysis. *Astronomy & Astrophysics*, 627:A21, 2019.

J. R. Percy. *Understanding variable stars*. Cambridge University Press, 2007.

Ó. Pimentel, P. A. Estévez, and F. Förster. Deep attention-based supernovae classification of multiband light curves. *The Astronomical Journal*, 165(1):18, 2022.

F. Rosenblatt. The perceptron - a perceiving and recognizing automaton. Technical Report 85-460-1, Cornell Aeronautical Laboratory, Ithaca, New York, January 1957.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

P. Sánchez-Sáez, I. Reyes, C. Valenzuela, F. Förster, S. Eyheramendy, F. Elorrieta, F. Bauer, G. Cabrera-Vives, P. Estévez, M. Catelan, et al. Alert classification for the alerce broker system: The light curve classifier. *The Astronomical Journal*, 161(3):141, 2021.

P. Shaw, J. Uszkoreit, and A. Vaswani. Self-attention with relative position representations. *arXiv preprint arXiv:1803.02155*, 2018.

R. W. Slawson, A. Prša, W. F. Welsh, J. A. Orosz, M. Rucker, N. Batalha, L. R. Doyle, S. G. Engle, K. Conroy, J. Coughlin, et al. Kepler eclipsing binary stars. ii. 2165 eclipsing binaries in the second data release. *The Astronomical Journal*, 142(5):160, 2011.

J. Su, Y. Lu, S. Pan, A. Murtadha, B. Wen, and Y. Liu. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021.

S. Sun, Z. Cao, H. Zhu, and J. Zhao. A survey of optimization methods from a machine learning perspective. *IEEE transactions on cybernetics*, 50(8):3668–3681, 2019.

I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014.

A. Udalski. The optical gravitational lensing experiment. real time data analysis systems in the ogle-iii survey. *arXiv preprint astro-ph/0401123*, 2004.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

B. Wang, L. Shang, C. Lioma, X. Jiang, H. Yang, Q. Liu, and J. G. Simonsen. On position embeddings in bert. In *International Conference on Learning Representations*, 2021.

Q. Wang, Y. Ma, K. Zhao, and Y. Tian. A comprehensive survey of loss functions in machine learning. *Annals of Data Science*, pages 1–26, 2020.

C. Yu, K. Li, Y. Zhang, J. Xiao, C. Cui, Y. Tao, S. Tang, C. Sun, and C. Bi. A survey on machine learning based light curve analysis for variable astronomical sources. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 11(5):e1425, 2021.

Appendix A

Methodology Figures

Different samples from each of the datasets can be seen in the figures of this chapter. Each sampled light curve displays the ID, the class to which it belongs, and the median cadence in days. In general, these Figures visually complement the explanations and distributions of the data shown in Chapter 3.

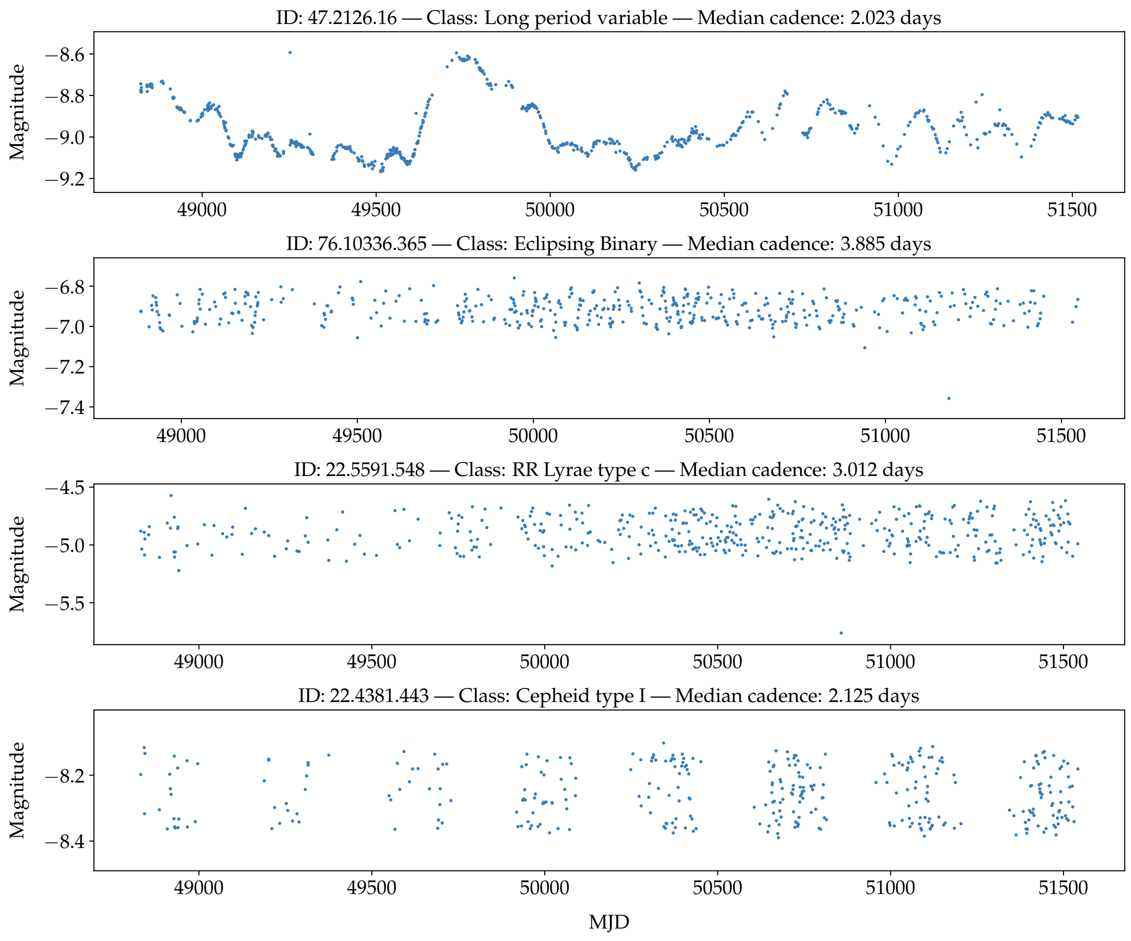

Figure A.1: Different samples from the dataset labeled MACHO.

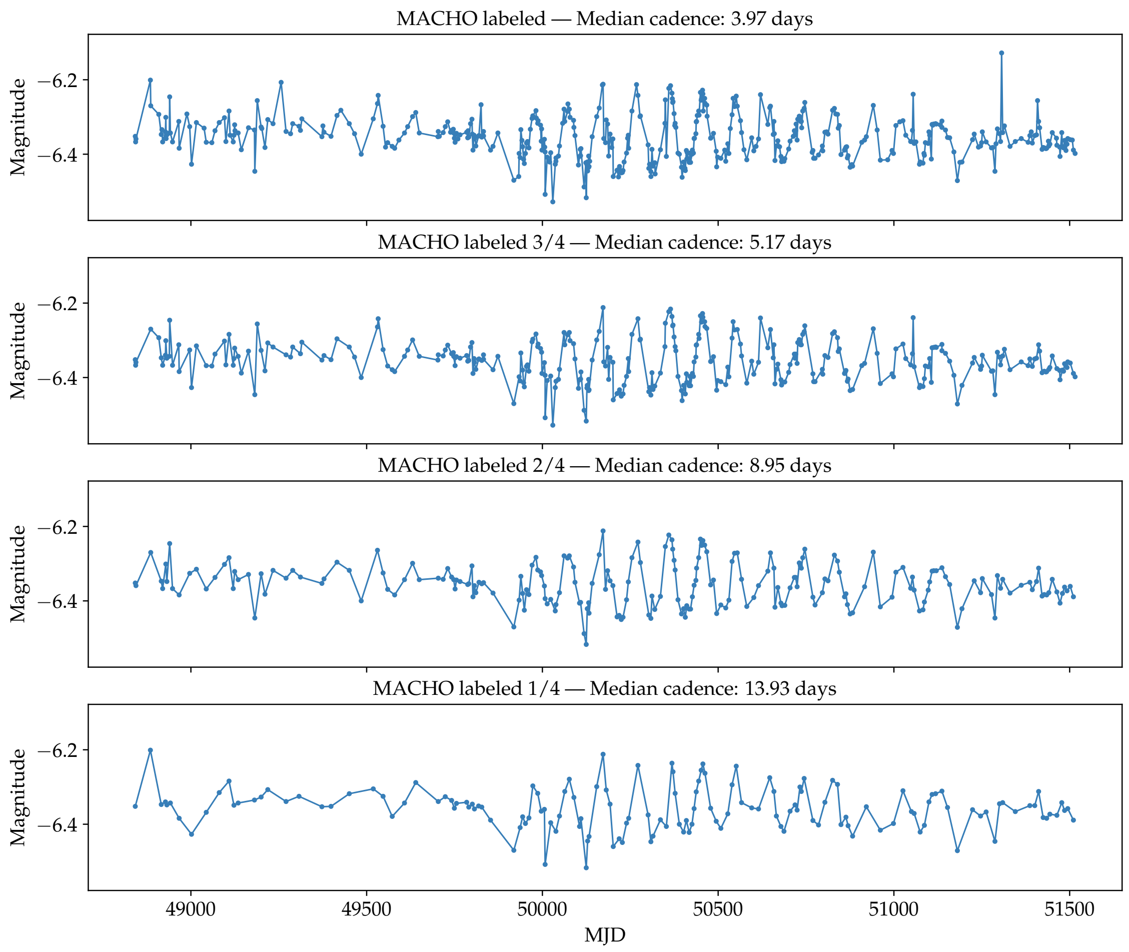

Figure A.2: A sample of a light curve with observations removed at varying rates from the labeled MACHO dataset.

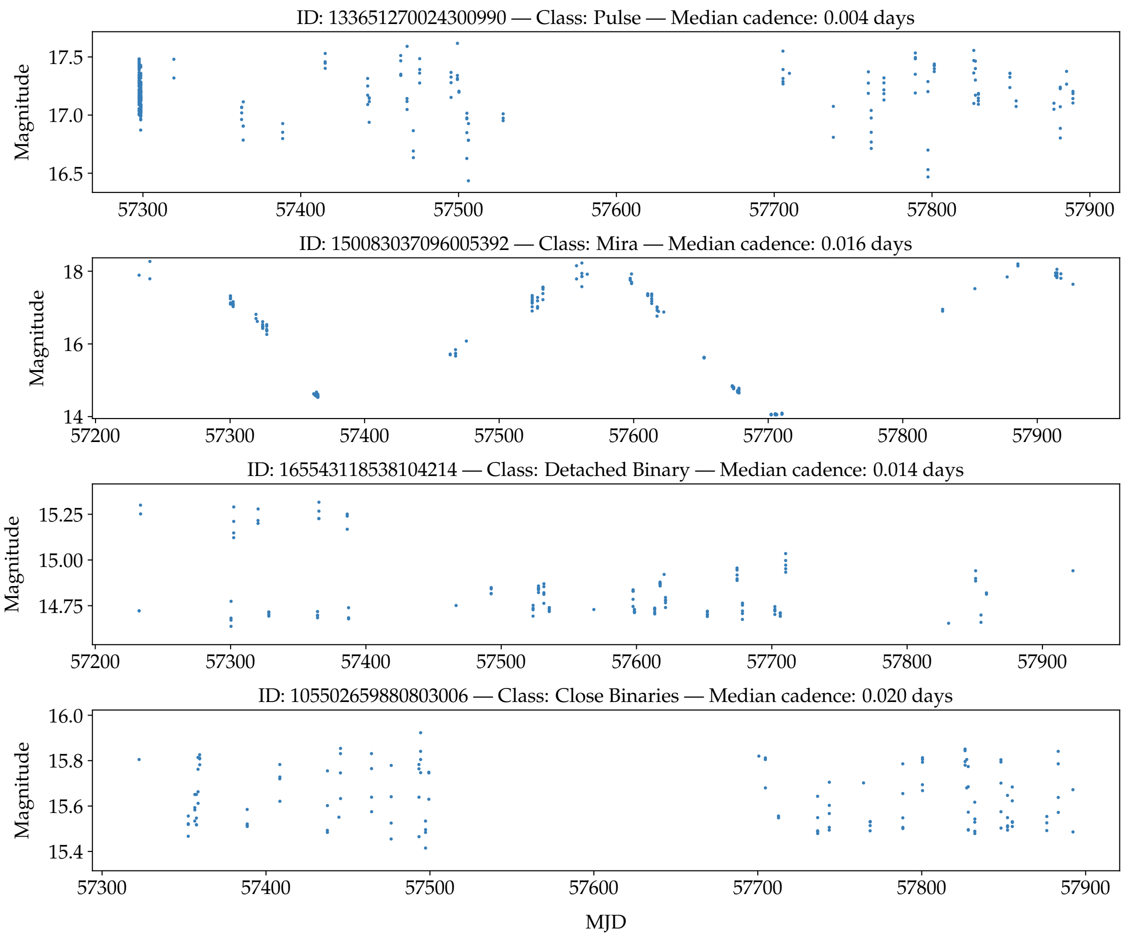

Figure A.3: Different samples from the dataset ATLAS.

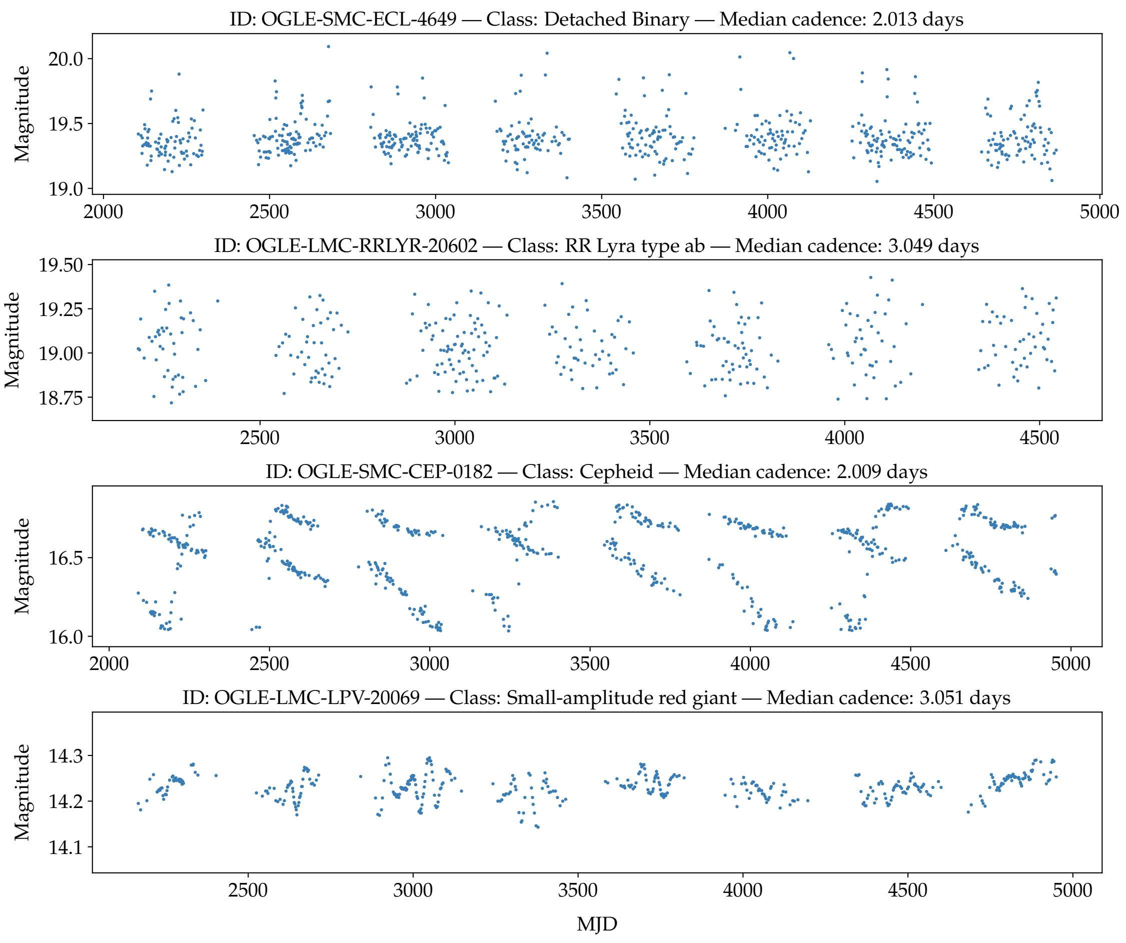

Figure A.4: Different samples from the dataset OGLE.

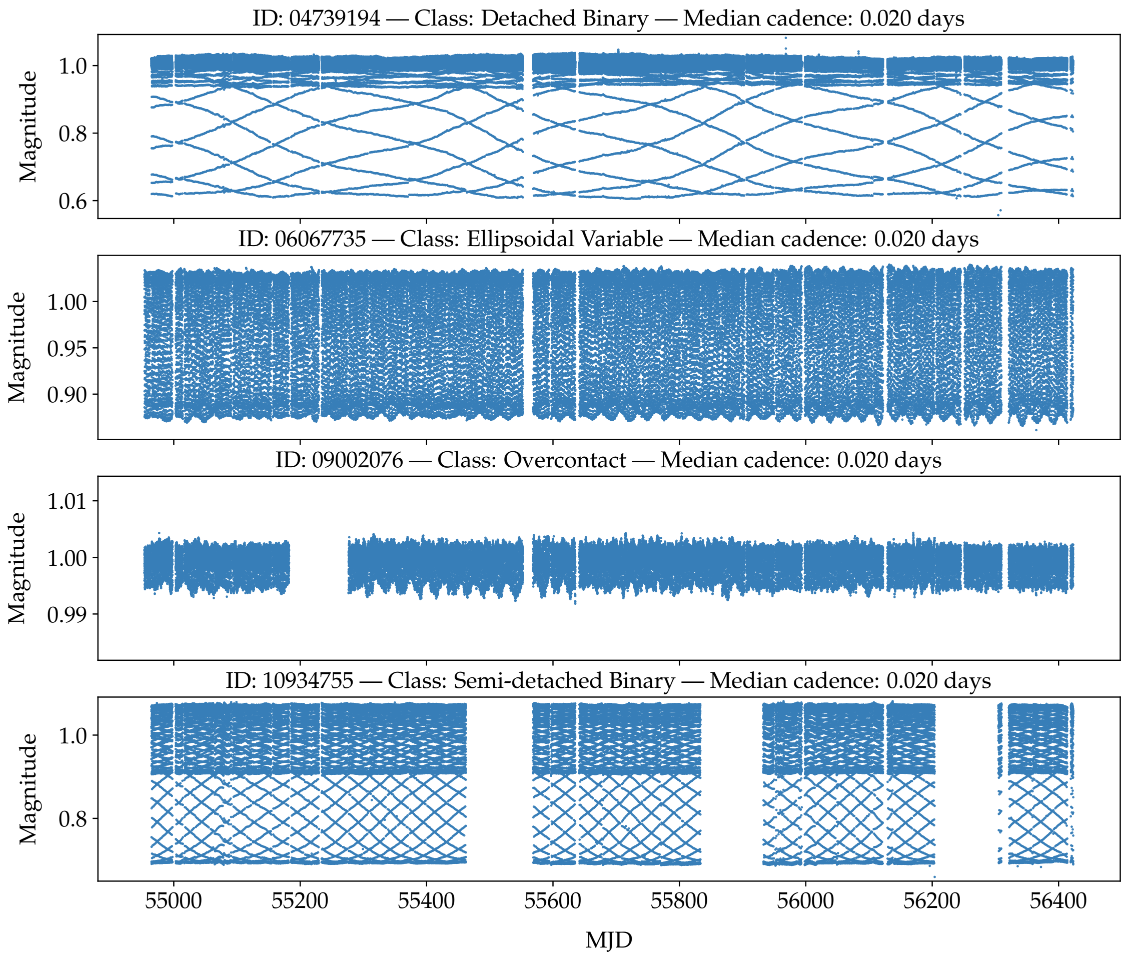

Figure A.5: Different samples from the dataset Kepler.

Appendix B

Results and Discussion Figures

In this chapter, we present the learning curves on the validation set for the proposed PEA and the positional encodings that achieved superior RMSE during pretraining. Figure B.1 illustrates the pretraining of transformers using the same hyperparameters. The y-axis represents the mean value of RMSE with a 4-step window, and the x-axis represents the number of epochs displayed on a logarithmic scale. It is evident that trainable positional encodings such as Trainable, Fourier, and Concat PE achieved comparable RMSE to the baseline with significantly fewer epochs in pretraining (38.6%, 27.2%, and 45.3% of baseline epochs, respectively). In contrast, the PEA method initially obtained a higher RMSE than the baseline, but it achieved an average RMSE of 0.204 earlier than the baseline by utilizing only 57.7% of the epochs.

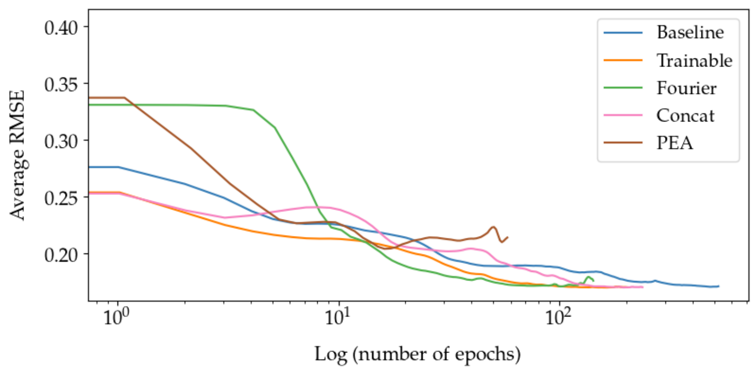

Figure B.1: Validation loss in pretraining stage.