The first step is the hardest: Pitfalls of Representing and Tokenizing Temporal Data for Large Language Models

Dimitris Spathis*, **Fahim Kawsar**†
Nokia Bell Labs
Cambridge, UK

Abstract

Objectives: Large Language Models (LLMs) have demonstrated remarkable generalization and across diverse tasks, leading individuals to increasingly use them as personal assistants due to their emerging reasoning capabilities. Nevertheless, a notable obstacle emerges when including numerical/temporal data into these prompts, such as data sourced from wearables or electronic health records. LLMs employ tokenizers in their input that break down text into smaller units. However, tokenizers are *not* designed to represent numerical values and might struggle to understand repetitive patterns and context, treating consecutive values as separate tokens and disregarding their temporal relationships. This paper discusses the challenges of representing and tokenizing temporal data. It argues that naively passing timeseries to LLMs can be ineffective due to the modality gap between numbers and text.
Materials and Methods: We conduct a case study by tokenizing a sample mobile sensing dataset using the OpenAI tokenizer. We also review recent works that feed timeseries data into LLMs for human-centric tasks, outlining common experimental setups like zero-shot prompting and few-shot learning.
Results: The case study shows that popular LLMs split timestamps and sensor values into multiple non-meaningful tokens, indicating they struggle with temporal data. We find that preliminary works rely heavily on prompt engineering and timeseries aggregation to "ground" LLMs, hinting that the "modality gap" hampers progress. The literature was critically analyzed through the lens of models optimizing for expressiveness versus parameter efficiency. On one end of the spectrum, training large domain-specific models from scratch is expressive but not parameter-efficient. On the other end, zero-shot prompting of LLMs is parameter-efficient but lacks expressiveness for temporal data.
Discussion: We argue tokenizers are not optimized for numerical data, while the scarcity of timeseries examples in training corpora exacerbates difficulties. We advocate balancing model expressiveness and computational efficiency when integrating temporal data. Prompt tuning, model grafting, and improved tokenizers are highlighted as promising directions.
Conclusion: We underscore that despite promising capabilities, LLMs cannot meaningfully process temporal data unless the input representation is addressed. We argue that this paradigm shift in how we leverage pre-trained models will particularly affect the area of biomedical signals, given the lack of modality-specific foundation models.

1 Introduction

In recent years, Large Language Models (LLMs) –also known as foundation models [1]– have garnered attention for their ability to generalize across a wide array of tasks. From natural language understanding to generating creative text, these models have demonstrated their prowess, often mimicking human-like performance. Unlike previous AI models, prompting enables example-based learning without any additional training (a concept known as in-context learning [2]). As their capabilities have grown, so too the idea of

*Also affiliated with the University of Cambridge. Correspondence: dimitrios.spathis@nokia.com
†Also affiliated with the University of Glasgow.

employing LLMs as general-purpose assistants has emerged, promising enhanced user experiences through personal devices.

However, amidst these advancements, a significant challenge arises when attempting to use existing LLMs with numerical and temporal data, particularly those originating from mobile sensors or medical timeseries. Perhaps overestimating LLMs' reasoning abilities, users are writing increasingly complex prompts combining e.g., text and data from lab tests or other measurements, expecting that models will be able to process them correctly. These data sources, often representing fine-grained behavioral and physiological states, introduce a complex layer of information that cannot be easily "translated" into text. LLMs, designed primarily for processing natural language, exhibit remarkable efficiency in representing textual input through specialized tokenizers, effectively dividing text into manageable units. Yet, as we start to integrate LLMs with temporal data, the complexities of this union become evident.

The tokenizers employed by LLMs appear to stumble when grappling with numerical inputs [3]. As we show below, repetitive patterns, an inherent characteristic of time-series data, can confound tokenizers, leading to the unintentional fragmentation of continuous sequences into disjointed tokens. Consequently, the temporal relationships that underpin such data may be lost in translation, potentially undermining the very essence of the information being processed.*

We argue that naively passing time series to LLMs can be ineffective because their input tokenizer layers are not optimized for numerical data. Therefore, we first show that such "zero-shot" approaches fail short due to the modality gap [4]† and advocate for different strategies. In this context, this paper delves into the nuances and obstacles that emerge when LLMs are confronted with the task of representing temporal data. We focus on the interplay between numerical and textual information, uncovering the potential pitfalls that can hamper the effective utilization of LLMs in scenarios where temporal context is important. We highlight that the key challenge for correctly processing such modalities into LLMs is the lack of available timeseries training data within their textual corpora. Last, we discuss potential solutions from the rapidly growing area of parameter-efficient transfer learning and multimodal adapters that could enable better integration of non-textual data into LLMs.

2 Tokenization in Language Models

Tokenization is a fundamental process underpinning the operation of LLMs. It involves the division of input and output texts into smaller, manageable units known as tokens. These tokens serve as the building blocks of language comprehension, enabling LLMs to process and generate text effectively. Tokens can encompass a variety of segments, such as characters, words, subwords, or symbols, depending on the chosen tokenization scheme.

The purpose of tokenization extends beyond mere text segmentation. It enables LLMs to handle diverse languages, vocabularies, and formats while mitigating computational and memory demands. The quality and diversity of generated text are influenced by tokenization, as it shapes the meaning and context of individual tokens. Tokenization strategies vary and can be rule-based, statistical, or learnable in nature, adapting to the complexity and variability of the input texts.

One of the prominent subword tokenization methods used in Transformers, the architecture underlying many LLMs, is Byte-Pair Encoding (BPE) [5]. OpenAI, for instance, employs BPE in its GPT-based models, which include around 50,000 tokens. BPE operates by iteratively merging the most frequently occurring pairs of characters or bytes into a single token until a predefined number of tokens or vocabulary size is reached. This approach is especially useful for accommodating rare or previously unseen words, resulting in more compact and consistent representations of text.

*Arguably, language also presents inherent short and long-term dependencies, however, given that LLMs are trained on large predominantly textual datasets, we expect that they learn more meaningful patterns between such tokens compared to numbers.

†We define "modality gap" as the challenge of mapping different modalities like text and signals in the same latent space. Existing works have shown that even in models whose modalities have been trained jointly, they reside in completely separate regions of the embedding space.

BPE variants include WordPiece [6] and SentencePiece [7]. WordPiece tokenization segments text into words or subwords, often leading to smaller vocabularies and efficient processing. SentencePiece, on the other hand, extends tokenization to the sentence level, empowering models to handle morphologically rich languages with complex sentence structures, such as Chinese and Japanese. Recent state-of-the-art models like Llama 2 employ BPE and SentencePiece by splitting all numbers into individual digits and use bytes to decompose unknown UTF-8 characters [8].

As an example, let's consider the word `"playing"` to illustrate how BPE works. First, the algorithm identifies the most frequent pair of characters, that according to our hypothetical corpus is `"ay"`. It then merges `"ay"` to create a new token: `"pl"+"ay"= "play"`. Our new updated vocabulary now is `"play"` and `"ing"`. In this example, by breaking down words into subword units and merging frequent pairs, BPE captures meaningful components of words, even those that might not have appeared in the original vocabulary.

3 Where do tokenizers struggle?

Despite its importance, tokenization introduces various challenges and open problems that impact the quality, interpretation, and usability of LLMs' outputs. While tokenization schemes like Byte-Pair Encoding (BPE) have been successful in segmenting text into meaningful units, several issues persist, highlighting areas where further research is needed.

Case sensitivity. Tokens are assigned numerical identifiers within a tokenizer's vocabulary. However, cases of words are treated as separate tokens, leading to inconsistencies in representation. For instance, `"good"` and `"Good"` are encoded as distinct tokens, hampering the LLM's ability to understand text, particularly in scenarios where capitalization matters.

Trailing whitespace. Tokens with whitespace are treated as separate entities; for instance, tokens such as `"the first step is "` and `"the first step is"` exhibit different representations, impacting the probabilities of subsequent tokens. This behavior introduces subtle biases that affect the model's language generation process and response characteristics.

Digit chunking. The tokenization of numerical values poses challenges, particularly when digits are inconsistently chunked. Numbers like `"480"` might be tokenized as a single unit, while `"481"` and `"482"` are split into two tokens. This inconsistency not only hampers the model's mathematical capabilities but also introduces complexities when dealing with consecutive values and temporal dependencies.

```
Input         → Token IDs
480, 481, 482 → 22148, 11, 4764, 16, 11, 4764, 17
```

Integers. The particular case of integer tokenization was investigated in a recent work [9]. In a series of simulations with popular LLMs, it was found that the tokenization of integers lacks a coherent decimal representation, leading to a fragmented approach where even basic mathematical operations require memorization rather than algorithmic processing. On the other hand, non-unique integers are tokenized through arbitrary and inconsistent chunking, complicating multi-digit arithmetic operations. The irregular division of numbers into tokens varies even for adjacent integers, demanding models to memorize arbitrary tokenization patterns. Notably, tokenization is influenced by the temporal distribution of data. Common dates, mostly in the 20th century, are assigned unique tokens, revealing the problematic interplay between tokenization and the characteristics of the training dataset.

Floating-point Numbers. These numbers, often representing decimal values, can be challenging to tokenize consistently. The representation of floating-point numbers can vary based on factors such as precision, scientific notation, and rounding errors. Tokenization of floating-point numbers involves decisions on whether to treat them as a single token or break them down into constituent parts. For instance, consider

the number `"3.14159"`, which might be tokenized as four different chunks by popular LLMs: `"3" + "." + "14" + "159"`.

```
Input   → Token IDs
3.14159 → 18, 13, 1415, 19707
```

Arithmetic reasoning. LLMs exhibiting inconsistent tokenization of numbers struggle with addition tasks that involve two-digit numbers and completely falter in adding larger numbers [10]. Specifically, their accuracy drops to zero for digits totalling five or more. This deficiency is attributed to the absence of a systematic approach to tokenizing individual digits. For instance, `"1234"` might be segmented into `"12"`, `"34"`, while `"5678"` could be split into `"5"` and `"678"`. Consequently, the model must grasp that a token's embedding could signify either a single digit or multiple digits, introducing complexity when mapping embeddings to varying-digit numbers. This irregular mapping challenge hinders the model's ability to accurately establish associations between embeddings and numbers of varying digit lengths.

Model-specific behavior. Tokenization is not universally consistent across different language models. Even when employing the same underlying method, different models can have distinct token representations. This model-specific nature of tokenization poses challenges for interoperability, pre-processing, and multi-modal modeling. It necessitates careful consideration when assessing various LLMs for real-world applications.

4 A case study on tokenizing temporal data

To showcase some of these challenges in the real world, we encode the first 20 samples of a popular mobile sensing dataset (WISDM [11]) via the OpenAI Tokenizer that powers ChatGPT [‡]. Each row contains the Participant ID, the activity label code, a UNIX timestamp, and the 3 accelerometer axes (x, y, z). As is common, models learn the temporal dependencies across the 3 axes in order to predict whether users perform activities such as running or walking. Activity recognition is a fundamental task in personal devices that, for example, can detect falls and notify one's family or emergency services. For illustration purposes, we choose not to filter or pre-process any information of the dataset.

In Figure 1, we show the raw input data along with the tokenized output and the token IDs. Affirming the concerns outlined in the previous section, we note the model's tendency to divide timestamps into multiple tokens, potentially rendering temporal dependencies across different rows irrelevant. With regard to the actual sensor data, each accelerometer value, which is naturally represented as a floating point number, is also split into multiple tokens. Adding to the complexity, the negative sign in front of the sensor values is grouped with the comma separator, thereby disregarding the direction of the sensor reading.

We argue that this representation cannot allow for any meaningful inference from LLMs. As we discuss in the following section, all existing preliminary works that attempt such inferences, have resorted to extensive filtering, such as downsampling, aggregating the signal, as well as rounding. As a result, raw data is rarely provided without context to models, with a lot of effort put into constructing well-crafted textual templates (prompts), in order to overcome this modality mismatch.

5 Large language models and timeseries

Researchers have started experimenting with treating LLMs as universal pre-trained models [12], despite any modality gap. In this section, we will discuss the motivation behind the need to incorporate temporal

[‡]The public OpenAI tokenizer was used in all examples in sections 3 and 4: https://platform.openai.com/tokenizer. The following file from user's 1600 smartwatch in WSDM dataset was used: `data_1600_accel_watch.txt`.

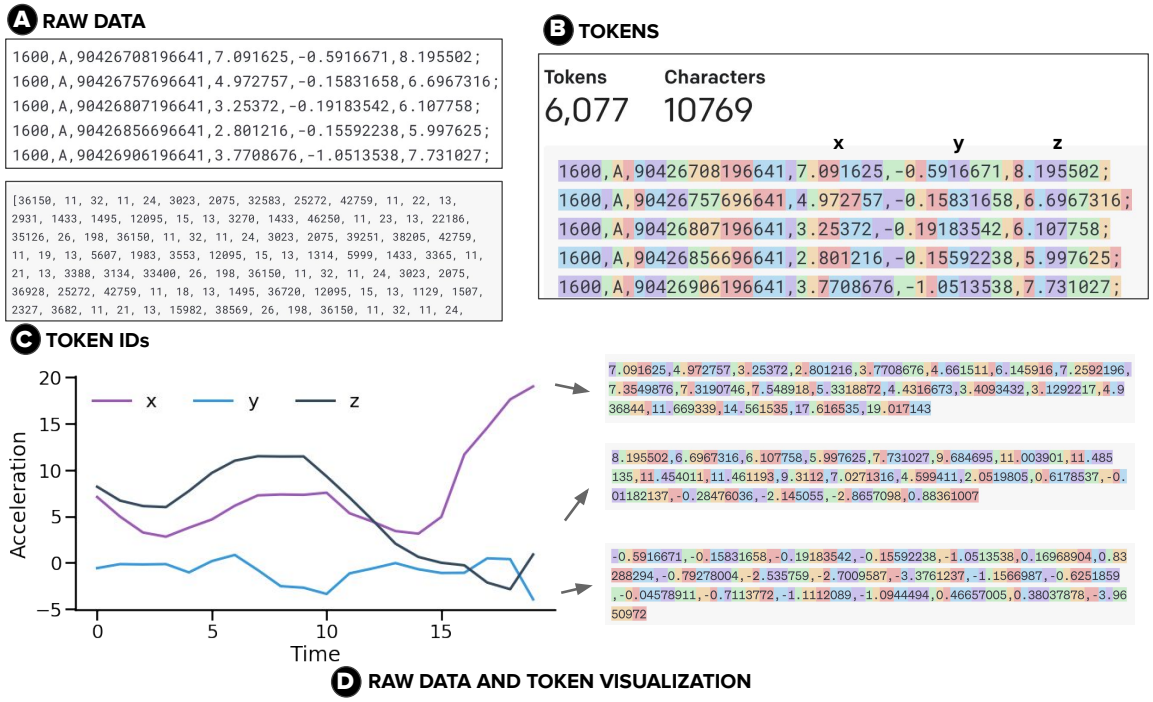

Figure 1: **Example of incorrect tokenization of a mobile sensing dataset**. Top panel: each row includes the Participant ID, activity label, UNIX timestamp, and x, y, z acceleration. The first few samples (a) of WISDM activity recognition dataset as it is processed into tokens (b). The timestamps, acceleration values, and negative signs are split into subtokens (c) that are not meaningful to models. Bottom panel: the acceleration values of each axis (x, y, z) as processed individually by the tokenizer (d). The different colours denote different token splits and might be repeated whereas the corresponding tokens are unique.

data to LLMs, common scenarios, and recent works on timeseries tasks, with a focus on high-frequency temporal data such as those found in biomedical signals. We also note that while there are plenty of works that borrow the Transformer/GPT architecture and re-train from scratch [13, 14], *here we focus on the paradigm of directly feeding temporal data to pre-trained LLMs.* Figure 2 provides a conceptual comparison between the key methods we discuss below, focusing on the tradeoff between computational efficiency (adaptation) and advanced reasoning capability (expressiveness).

Why would we pass time series into LLMs? LLMs are predominantly trained on large text datasets crawled from the public web. Datasets such as The Pile and Common Crawl [15] contain hundreds of gigabytes of text sourced from Wikipedia, Google Patents, Reddit, and other places. While the exact percentage of text versus other types of data such as numbers is not explicitly mentioned in the available sources, we expect that the available numerical information is limited. As a result, the most frequent tokens dominate the rest. Additionally, due to the inherent nature of language being a discrete medium, compared to numbers being continuous, there are considerably more variations in the latter. On the other hand, training foundation models on timeseries data only might seem like a promising direction, especially in cases we are interested in accuracy over reasoning capabilities. Attempts like TimeCLR [16] or Apple's AHMS [17] combine multiple timeseries datasets from appropriate repositories like the UCR Archive or health studies, which are, however, magnitudes smaller than text datasets. The above also motivates the adaptation of LLMs over modality-specific models.

Despite the popularity of LLMs in language tasks, the established modeling paradigm of using neural networks for timeseries tasks still involves a combination of convolutional, recurrent, and attention layers [18]. For example, in activity recognition or arrhythmia detection, a common workflow involves a sliding window which splits the input signal into fixed-sized sequences of channels (accelerometer axes or ECG

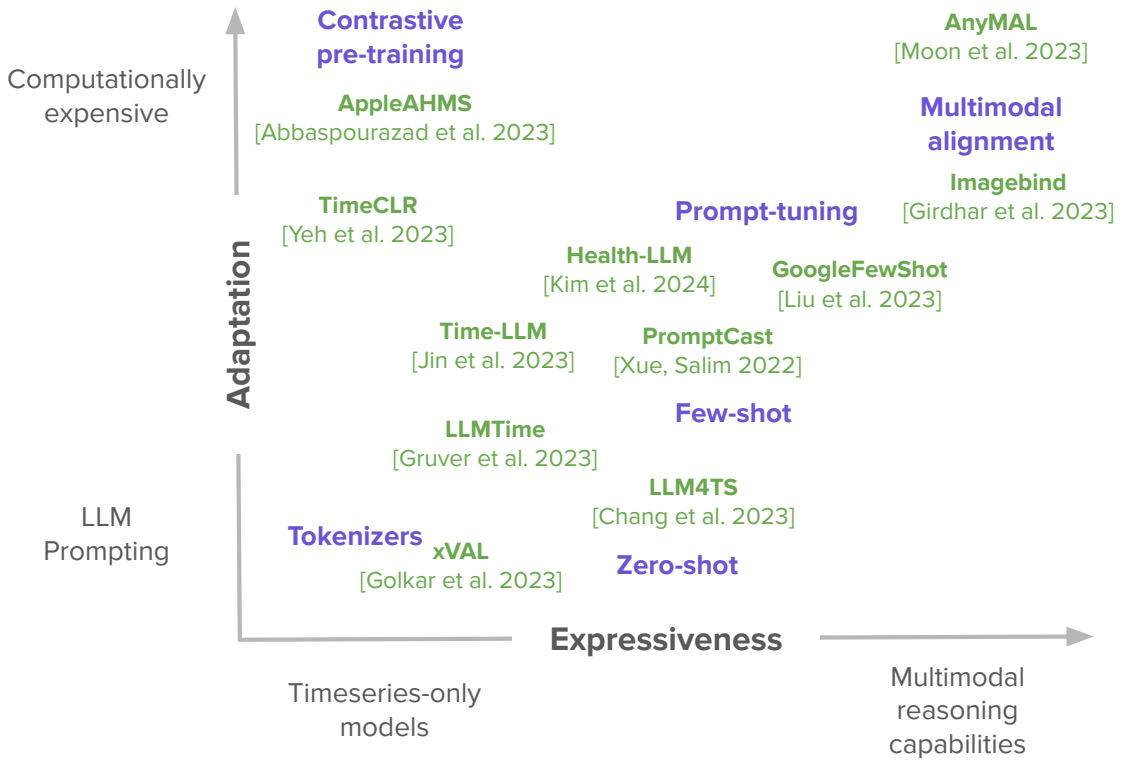

Figure 2: **Conceptual comparison of proposed methods for timeseries-aware LLMs seen as a function of expressiveness and adaptation/computational requirements**. The main paradigms are listed in purple alongside some key works in the area in green. We notice four distinct areas: pre-training from scratch using domain data, new tokenizers and prompting methods which require minimal fine-tuning, light-weight adaptation like prompt-tuning that try to balance both dimensions, and multimodal adapters that provide more reasoning capabilities at the cost of aligning every modality. There is a noticeable gap in the lower right quadrant showcasing the tradeoff between parameter efficiency and advanced reasoning. We note that the exact position of each method is based on a qualitative assessment of the overall contribution and should not be used for individual comparisons (similar works are "jittered"); we recommend interpreting this figure on the quadrant level.

leads, respectively). Due to inherent difficulties in collecting large labeled datasets, self-supervised learning has proved to be particularly effective in such tasks [19, 20, 21, 22]. However, even large *unlabeled* data of that kind is hard to encounter because it is not publicly available on the web, unlike images or text. Another crucial limiting factor is the number of potential modalities; for example, aligned and paired data of movement, heart rate, audio, and sleep patterns might be impossible to collect and share at scale due to privacy concerns. As a result, available pre-trained models in areas such as physical activity or heart health are limited in size and generalization capabilities, compared to popular foundation models [17, 23, 24]. This further motivates the shift from scarce domain-specific models to repurposing general foundation models and LLMs.

Feeding timeseries to LLMs. Following the established experimental setup from the NLP community, the following evaluation protocols are usually followed when feeding timeseries data to LLMs:

- **Zero-shot inference [25]**: The input is either the raw timeseries or its combination with associated text that might describe the data. The LLM makes predictions directly on the test data without any additional tuning or training. This tests the out-of-the-box capabilities of the model but most usually performance is poor and hence it is used as a "lower bound" baseline. Especially in cases where the numerical data dominate the input, the model cannot "ground" the data to any language patterns.

- **Prompt engineering [26]**: Building on the textual component of the previous scenario, prompt engineering refers to methods that enrich the text accompanying the signals. Enrichment methods include adding context such as additional metadata, health context, temporal context, and combinations thereof. This guides the model to the potential "solution space" because it helps to contextualize the raw signals. As in zero-shot mode, it operates on test-time and does not update any model parameters. The two methods can also be combined.

- **Few-shot/In-context learning [27]** : While zero-shot inference involves single inputs without any associated labels or answers, few-shot or in-context learning uses a limited selection of demonstrations within the prompts and in contrast to the previous approaches, this requires pairs of <data, label>, however, the model parameters still stay fixed. Few-shot learning can be used together with prompt engineering.

- **Fine-tuning [28]**: This approach involves a partial re-training of the model parameters and as such it is the most computationally expensive of the four. A trainable embedding is added to the input of the LLM (prompt-tuning) or the intermediate layers (prefix-tuning). While fine-tuning helps to specialize the model to each respective task, it might result in forgetting more general knowledge. We delve into the tradeoffs of fine-tuning in section 6.

Applications to human-centric signals. Recent works have applied the above scenarios to physiological signals, human gestures and mobility data. These initial works have started investigating time series and LLMs for health, but they have limitations: they mostly rely on data curation or aggregation, handcrafted prompt-engineering, and proprietary models whose results cannot be reproduced independently. For instance, a pre-print by Liu et al. [29] (referred to as "GoogleFewShot" in Fig. 2) looked into heart rhythm classification, activity recognition, calorie estimation, and predicting stress levels and mental health scores from Fitbit data. The paper also explores the effects of providing textual context versus solely numerical data as input to the LLM. Across almost all tasks, including more contextual information like *"Classify the given Interbeat Interval sequence in ms"* led to improved performance over just feeding in the raw numbers while fine-tuning achieved the best performance. Regarding the base model, the authors employ a proprietary LLM (PaLM [30]) which uses the SentencePad tokenizer with a 256K-sized token vocabulary. While acknowledging the promise of these results, there is limited information on any pre-processing or tokenization. We assume that once in textual prompt format, the time-series data is directly fed into the base model without any further vectorization or embedding. In section 6, we expand on the importance of prompt tuning and compare it to other approaches that could alleviate modality mismatches between numbers and language.

Similar to the above study, another pre-print (Health-LLM) evaluated various physiological signals for the prediction of stress, physical activities, calorie expenditure, sleep quality, and cardiovascular health [31]. This work focused on prompt enhancement by proposing multiple strategies to incorporate context into the prompts. These strategies included adding user metadata such as age or sex, health context such as definitions of the outcome at hand, and temporal context that summarized the timeseries in sentence. The combination of such methods yielded up to 23.8% improvement in performance in a fine-tuned model. In a medical context, the prompting enrichment approaches discussed in this work could provide better interpretability when audited by medical professionals, compared to adapter-based methods that map data to vectors.

Other preliminary works followed similar experimental setups. In a zero-shot scenario using a small-scale dataset, the popular GPT4 model was fed in hand gesture data through light and vibration sensors [32]. The authors report high performance using prompt engineering templates such as: *"I use an RGB Light and Proximity Sensor to detect the gestures Single air tap, Double air Tap, and Triple air Tap.* [..] *Given below are sample data collected at three instances for each gesture:* [*Numbers*]*"*.

PromptCast also followed a similar scenario by framing timeseries forecasting as a language generation problem and introduced a new prompt template dataset covering weather, energy, and human mobility forecasting [33]. Experiments compared state-of-the-art numerical forecasting methods like Informer, Autoformer, and FEDformer against language models like T5, BART, and BigBird. The results showed that language models are competitive to numerical methods when fine-tuned on the datasets, with Bigbird, BART,

and RoBERTa being the top performers. The language models also showed stronger generalization under zero-shot transfer. We attribute the success of such a setup to careful dataset creation and the short sequence length (15 timesteps) used in the prompts. We expect that by incorporating prompt-tuning or other adapter layers, these models would further improve over purely numerical models.

6 How to better integrate temporal data into LLMs?

Despite the encouraging results, we see that these preliminary works have to "ground" the LLMs in numerical data using verbose hand-engineered prompts and extensive aggregation of timeseries. In other words, they rely predominantly on the power of text to contextualize numerical data. This paper argues that we will only unlock the real value of temporal data of that kind, once we map it to a more meaningful representation. While modality mappings between images and text are well-studied [4, 34], the link of timeseries to text remains virtually unexplored. We point the reader to Jin et al. [35] for a comprehensive view of LLMs for timeseries. In this section, we highlight the following four key research directions towards better integrating time series into language models: prompt-tuning, model grafting, improved tokenizers, and hybrid approaches.

Prompt tuning. Parameter-Efficient Fine-Tuning (PEFT) has emerged as a potential solution to this problem. Traditional neural network fine-tuning would involve a pre-trained network that is fixed (or frozen) and new trainable layers that are added to the end of the network in order to tailor it to a specific task. Instead, PEFT involves prompting or lightweight layers that are added usually to the input (prompt-tuning) or selected layers throughout the model (prefix-tuning).

In general, prompt tuning refers to techniques that change the LLM prompt to achieve better results. Directly changing the input content is known as *hard* prompt. More interestingly, *soft* prompting involves prepending a trainable tensor to the input which can then be optimized to a downstream task via backpropagation [36]. As seen in the previous section, this approach is particularly effective because the prompt layer encodes information that assists the model in comprehending timeseries data that is absent in the original frozen LLM [29]. This is significantly more parameter-efficient than full fine-tuning which becomes prohibitive as the model size and the number of tasks grow.

On a similar note, modules known as *Adapters* insert small trainable blocks to each layer of the pre-trained network with only these blocks being fine-tuned. For example, a popular method, LoRa, injects trainable low-rank matrices into transformer layers to approximate the weight updates [37]. While these approaches have similar goals, here we are interested in prompt-tuning because it operates on the input space and can effectively steer the model to process timeseries data [§].

Model grafting. By viewing the mismatch between numbers and text as a multimodal understanding problem, the notion of model grafting has emerged as another solution [39]. For example, in a recently proposed model called HeLM, non-text modalities are mapped via encoders (trained over input examples) into the same token embedding space as text, where a separate encoder is learned for each modality [40]. In particular, HeLM trained two individual encoders to map spirogram sequences (lung capacity measurements) and demographic data into a limited-size token vocabulary that were inserted into the text tokens (being order-agnostic). Modality-specific encoders are also common in multimodal foundation models such as ImageBind [41], IMU2CLIP [42], AnyMAL [43], and Meta-Transformer [44], with the latter proposing a data-agnostic approach to merge token embeddings into a shared manifold space. Most notably, while approaches like Imagebind align every modality with vision, AnyMAL does so through language by leveraging more powerful and larger LLMs, hinting that language can be used as the universal proxy to other modalities. In the same vein, ELIXR [45], LLaVA [46], and BLIP2 [47] introduced adapters[¶] that mapped the activations of an image encoder into an LLM-understandable form.

Model grafting has both advantages and disadvantages. On the one hand, it is computationally efficient when

[§]A unified view across all tuning methods is provided in [38].

[¶]Term disambiguation: while adapters like LoRa insert trainable modules within layers, adapters like ELIXR correspond to independent models that plug into other pre-trained LLMs.

training the adapter layers, while allowing the LLM to connect to other high-performing models from different domains (e.g., a well-trained encoder of ECG data). Breaking down the problem into encoder, adapter, and LLM components can also facilitate faster testing and iteration cycles. On the other hand, this modularization introduces complexity by dealing with a system that is not trained end-to-end. Last, compared to natural language prompts, the communication between the specialist encoder (e.g., image or timeseries) and the LLM is no longer meaningful to humans since the mapping is represented as high-dimensional vectors [48].

Improved tokenizers. Besides training lightweight adapters, we can rethink the design of tokenizers for mixed textual and numerical data. LLaMa was recently shown to outperform GPT-4 in arithmetic tasks due to splitting each digit into an individual token, thereby ensuring consistent tokenization of numbers [49]. Specialized models trained on the scientific literature and notation also used similar digit-level splitting, pointing to improved understanding of symbolic and numerical data [50]. LLMTime [51] introduced a new tokenization method that separates the digits with spaces, drops decimal points, scales down values, and converts tokens into continuous values, enabling frozen LLMs to outperform domain-specific models. xVAL [52] focused on processing the numbers independently by storing each number in a separate vector, using a single token. Then, the pre-processed text is encoded into a finite series of word tokens, while the embeddings of the number tokens are multiplied with their corresponding numerical values.

Hybrid approaches. While all these approaches can be seen in isolation, some works have started incorporating them into single models. For instance, Time-LLM [53] uses a reprogramming [54] method to align timeseries and text clusters (prototypes) via patching, which acts as tokenization. Additional prompt prefixes such as trends and lags and other statistics are concatenated to the input as embeddings to enrich reasoning capabilities. Other works like the LLM4TS [55] used both alignment and fine-tuning but addressed the challenge of limited context window length by adopting channel-independence along with patching for time-series tokenization, by converting multivariate timeseries into multiple univariate timeseries.

7 Discussion

This paper discussed the emerging idea of incorporating temporal data into pre-trained LLMs and proposed solutions for better encoding of numerical inputs into LLM-understandable information. We argue that this paradigm shift on how we leverage pre-trained models will particularly affect the area of biomedical signals, given the lack of modality-specific foundation models. To bridge the "modality gap", practitioners should focus on methods that optimize for both expressiveness and parameter-efficiency such as model grafting with multimodal adapters, along with careful reassessment of existing tokenizers and embedders. Beyond tokenization, the next generation of LLMs should address other architectural constraints such as longer context windows, towards better handling of real-world temporal datasets. Last, while our scope is limited to temporal data, these approaches could generalize to other non-language tasks [56].

Disclosure Statements

Funding. This research received no specific grant from any funding agency in the public, commercial or not-for-profit sectors.
Competing Interests. The authors have no competing interests to declare.
Contributorship. D.S. wrote the first draft of the manuscript and produced the figures. All authors (D.S., F.K.) critically reviewed, contributed to the preparation of the manuscript, and approved the final version. All authors vouch for the data, analyses, and interpretations.
Data Availability. All datasets used in our analysis are available in previous works [11].

References

[1] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[2] OpenAI. Gpt-4 technical report. *arXiv*, pages 2303–08774, 2023.

[3] Subhabrata Mukherjee, Paul Gamble, Markel Sanz Ausin, Neel Kant, Kriti Aggarwal, Neha Manjunath, Debajyoti Datta, Zhengliang Liu, Jiayuan Ding, Sophia Busacca, et al. Polaris: A safety-focused llm constellation architecture for healthcare. *arXiv preprint arXiv:2403.13313*, 2024.

[4] Victor Weixin Liang, Yuhui Zhang, Yongchan Kwon, Serena Yeung, and James Y Zou. Mind the gap: Understanding the modality gap in multi-modal contrastive representation learning. *Advances in Neural Information Processing Systems*, 35:17612–17625, 2022.

[5] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *54th Annual Meeting of the Association for Computational Linguistics*, pages 1715–1725. Association for Computational Linguistics (ACL), 2016.

[6] Mike Schuster and Kaisuke Nakajima. Japanese and korean voice search. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5149–5152. IEEE, 2012.

[7] Taku Kudo and John Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*, 2018.

[8] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[9] Beren Millidge. Integer tokenization is insane — beren.io. https://www.beren.io/2023-02-04-Integer-tokenization-is-insane/, 2023. [Accessed 18-08-2023].

[10] Rodrigo Nogueira, Zhiying Jiang, and Jimmy Lin. Investigating the limitations of transformers with simple arithmetic tasks. *arXiv preprint arXiv:2102.13019*, 2021.

[11] Gary M Weiss, Kenichi Yoneda, and Thaier Hayajneh. Smartphone and smartwatch-based biometrics using activities of daily living. *IEEE Access*, 7:133190–133202, 2019.

[12] Kevin Lu, Aditya Grover, Pieter Abbeel, and Igor Mordatch. Pretrained transformers as universal computation engines. *arXiv preprint arXiv:2103.05247*, 1, 2021.

[13] Akhil Vaid, Joy Jiang, Ashwin Sawant, Stamatios Lerakis, Edgar Argulian, Yuri Ahuja, Joshua Lampert, Alexander Charney, Hayit Greenspan, Jagat Narula, et al. A foundational vision transformer improves diagnostic performance for electrocardiograms. *NPJ Digital Medicine*, 6(1):108, 2023.

[14] Pierre Louis Gaudilliere, Halla Sigurthorsdottir, Clémentine Aguet, Jérôme Van Zaen, Mathieu Lemay, and Ricard Delgado-Gonzalo. Generative pre-trained transformer for cardiac abnormality detection. In *2021 Computing in Cardiology (CinC)*, volume 48, pages 1–4. IEEE, 2021.

[15] Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

[16] Chin-Chia Michael Yeh, Xin Dai, Huiyuan Chen, Yan Zheng, Yujie Fan, Audrey Der, Vivian Lai, Zhongfang Zhuang, Junpeng Wang, Liang Wang, et al. Toward a foundation model for time series data. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 4400–4404, 2023.

[17] Salar Abbaspourazad, Oussama Elachqar, Andrew C. Miller, Saba Emrani, Udhyakumar Nallasamy, and Ian Shapiro. Large-scale training of foundation models for wearable biosignals, 2023.

[18] Haojie Ma, Wenzhong Li, Xiao Zhang, Songcheng Gao, and Sanglu Lu. Attnsense: Multi-level attention mechanism for multimodal human activity recognition. In *IJCAI*, pages 3109–3115, 2019.

[19] Aaqib Saeed, Tanir Ozcelebi, and Johan Lukkien. Multi-task self-supervised learning for human activity detection. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 3(2):1–30, 2019.

[20] Harish Haresamudram, Irfan Essa, and Thomas Plötz. Assessing the state of self-supervised human activity recognition using wearables. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 6(3):1–47, 2022.

[21] Chi Ian Tang, Ignacio Perez-Pozuelo, Dimitris Spathis, Soren Brage, Nick Wareham, and Cecilia Mascolo. Selfhar: Improving human activity recognition through self-training with unlabeled data. *Proceedings of the ACM on interactive, mobile, wearable and ubiquitous technologies*, 5(1):1–30, 2021.

[22] Dimitris Spathis, Ignacio Perez-Pozuelo, Laia Marques-Fernandez, and Cecilia Mascolo. Breaking away from labels: The promise of self-supervised machine learning in intelligent health. *Patterns*, 3(2), 2022.

[23] Hang Yuan, Shing Chan, Andrew P Creagh, Catherine Tong, David A Clifton, and Aiden Doherty. Self-supervised learning for human activity recognition using 700,000 person-days of wearable data. *arXiv preprint arXiv:2206.02909*, 2022.

[24] Dimitris Spathis, Ignacio Perez-Pozuelo, Soren Brage, Nicholas J Wareham, and Cecilia Mascolo. Self-supervised transfer learning of physiological representations from free-living wearable data. In *Proceedings of the Conference on Health, Inference, and Learning*, pages 69–78, 2021.

[25] Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

[26] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35, 2023.

[27] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[28] Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235, 2023.

[29] Xin Liu, Daniel McDuff, Geza Kovacs, Isaac Galatzer-Levy, Jacob Sunshine, Jiening Zhan, Ming-Zher Poh, Shun Liao, Paolo Di Achille, and Shwetak Patel. Large language models are few-shot health learners. *arXiv preprint arXiv:2305.15525*, 2023.

[30] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

[31] Yubin Kim, Xuhai Xu, Daniel McDuff, Cynthia Breazeal, and Hae Won Park. Health-llm: Large language models for health prediction via wearable sensor data. *arXiv preprint arXiv:2401.06866*, 2024.

[32] Pramuka Medaranga Sooriya Patabandige, Steven Antya Orvala Waskito, Kunjun Li, Kai Jie Leow, Shantanu Chakrabarty, and Ambuj Varshney. Poster: Rethinking embedded sensor data processing and analysis with large language models. In *Proceedings of the 21st Annual International Conference on Mobile Systems, Applications and Services*, pages 561–562, 2023.

[33] Hao Xue and Flora D Salim. Prompt-based time series forecasting: A new task and dataset. *arXiv preprint arXiv:2210.08964*, 2022.

[34] Peiyang Shi, Michael C Welle, Mårten Björkman, and Danica Kragic. Towards understanding the modality gap in clip. In *ICLR 2023 Workshop on Multimodal Representation Learning: Perks and Pitfalls*, 2023.

[35] Ming Jin, Yifan Zhang, Wei Chen, Kexin Zhang, Yuxuan Liang, Bin Yang, Jindong Wang, Shirui Pan, and Qingsong Wen. Position paper: What can large language models tell us about time series analysis, 2024.

[36] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

[37] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[38] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*, 2021.

[39] Zewei Sun, Mingxuan Wang, and Lei Li. Multilingual translation via grafting pre-trained language models. *arXiv preprint arXiv:2109.05256*, 2021.

[40] Anastasiya Belyaeva, Justin Cosentino, Farhad Hormozdiari, Cory Y McLean, and Nicholas A Furlotte. Multimodal llms for health grounded in individual-specific data. *arXiv preprint arXiv:2307.09018*, 2023.

[41] Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. Imagebind: One embedding space to bind them all. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15180–15190, 2023.

[42] Seungwhan Moon, Andrea Madotto, Zhaojiang Lin, Alireza Dirafzoon, Aparajita Saraf, Amy Bearman, and Babak Damavandi. Imu2clip: Multimodal contrastive learning for imu motion sensors from egocentric videos and text. *arXiv preprint arXiv:2210.14395*, 2022.

[43] Seungwhan Moon, Andrea Madotto, Zhaojiang Lin, Tushar Nagarajan, Matt Smith, Shashank Jain, Chun-Fu Yeh, Prakash Murugesan, Peyman Heidari, Yue Liu, et al. Anymal: An efficient and scalable any-modality augmented language model. *arXiv preprint arXiv:2309.16058*, 2023.

[44] Yiyuan Zhang, Kaixiong Gong, Kaipeng Zhang, Hongsheng Li, Yu Qiao, Wanli Ouyang, and Xiangyu Yue. Meta-transformer: A unified framework for multimodal learning. *arXiv preprint arXiv:2307.10802*, 2023.

[45] Shawn Xu, Lin Yang, Christopher Kelly, Marcin Sieniek, Timo Kohlberger, Martin Ma, Wei-Hung Weng, Attila Kiraly, Sahar Kazemzadeh, Zakkai Melamed, Jungyeon Park, Patricia Strachan, Yun Liu, Chuck Lau, Preeti Singh, Christina Chen, Mozziyar Etemadi, Sreenivasa Raju Kalidindi, Yossi Matias, Katherine Chou, Greg S. Corrado, Shravya Shetty, Daniel Tse, Shruthi Prabhakara, Daniel Golden, Rory Pilgrim, Krish Eswaran, and Andrew Sellergren. Elixr: Towards a general purpose x-ray artificial intelligence system through alignment of large language models and radiology vision encoders, 2023.

[46] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*, 2023.

[47] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*, 2023.

[48] Greg Corrado and Yossi Matias. Multimodal medical AI — ai.googleblog.com. https://ai.googleblog.com/2023/08/multimodal-medical-ai.html, 2023. [Accessed 18-08-2023].

[49] Tiedong Liu and Bryan Kian Hsiang Low. Goat: Fine-tuned llama outperforms gpt-4 on arithmetic tasks, 2023.

[50] Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. Galactica: A large language model for science, 2022.

[51] Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew Gordon Wilson. Large language models are zero-shot time series forecasters. *arXiv preprint arXiv:2310.07820*, 2023.

[52] Siavash Golkar, Mariel Pettee, Michael Eickenberg, Alberto Bietti, Miles Cranmer, Geraud Krawezik, Francois Lanusse, Michael McCabe, Ruben Ohana, Liam Parker, et al. xval: A continuous number encoding for large language models. *arXiv preprint arXiv:2310.02989*, 2023.

[53] Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, et al. Time-llm: Time series forecasting by reprogramming large language models. *arXiv preprint arXiv:2310.01728*, 2023.

[54] Gamaleldin F Elsayed, Ian Goodfellow, and Jascha Sohl-Dickstein. Adversarial reprogramming of neural networks. *arXiv preprint arXiv:1806.11146*, 2018.

[55] Ching Chang, Wen-Chih Peng, and Tien-Fu Chen. Llm4ts: Two-stage fine-tuning for time-series forecasting with pre-trained llms. *arXiv preprint arXiv:2308.08469*, 2023.

[56] Tuan Dinh, Yuchen Zeng, Ruisu Zhang, Ziqian Lin, Michael Gira, Shashank Rajput, Jy-yong Sohn, Dimitris Papailiopoulos, and Kangwook Lee. Lift: Language-interfaced fine-tuning for non-language machine learning tasks. *Advances in Neural Information Processing Systems*, 35:11763–11784, 2022.