Proceedings of Machine Learning Research LEAVE UNSET:1–24, 2025 Submitted LEAVE UNSET; Published LEAVE UNSET

ECG-Byte: A Tokenizer for End-to-End Generative Electrocardiogram Language Modeling

William Jongwon Han wjhan@andrew.cmu.edu
Carnegie Mellon University, USA

Chaojing Duan chaojing.duan@ahn.org
Allegheny Health Network, USA

Michael A. Rosenberg michael.a.rosenberg@cuanschutz.edu
University of Colorado, USA

Emerson Liu emerson.liu@ahn.org
Allegheny Health Network, USA

Ding Zhao dingzhao@andrew.cmu.edu
Carnegie Mellon University, USA

Abstract

Large Language Models (LLMs) have shown remarkable adaptability across domains beyond text, specifically electrocardiograms (ECGs). More specifically, there is a growing body of work exploring the task of generating text from a multi-channeled ECG and corresponding textual prompt. Current approaches typically involve pretraining an ECG-specific encoder with a self-supervised learning (SSL) objective and using the features output by the pretrained encoder to finetune a LLM for natural language generation (NLG). However, these methods are limited by 1) inefficiency from two-stage training and 2) interpretability challenges with encoder-generated features. To address these limitations, we introduce **ECG-Byte**, an adapted byte pair encoding (BPE) tokenizer pipeline for autoregressive language modeling of ECGs. This approach compresses and encodes ECG signals into tokens, enabling end-to-end LLM training by combining ECG and text tokens directly, while being much more interpretable since the ECG tokens can be directly mapped back to the original signal. Using **ECG-Byte**, we achieve competitive performance in NLG tasks in only *half* the time and ∼48% of the data required by two-stage approaches.

Data and Code Availability This paper uses the ECG-Chat pretraining (Zhao et al., 2024) and ECG-QA datasets (Oh et al., 2023), which were both created from the MIMIC-IV ECG (Johnson et al., 2023) and PTB-XL datasets (Wagner et al., 2020). More details about the datasets are provided in Section 3. The ECG-Chat pretraining, ECG-QA, MIMIC-IV ECG, and PTB-XL datasets are all freely available on https://github.com/YubaoZhao/ECG-Chat, https://github.com/Jwoo5/ecg-qa, and https://physionet.org/ respectively. Lastly, we have released the code at the following link: https://github.com/willxxy/ECG-Byte.

Institutional Review Board (IRB) All datasets used in this study are directly taken from the publicly available, de-identified MIMIC-IV ECG (Johnson et al., 2023) and PTB-XL (Wagner et al., 2020) datasets, thus not requiring IRB approval.

1. Introduction

Cardiovascular diseases (CVDs) are the leading cause of global mortality, with 17.9 million lives taken each year and increasing (Organization, 2024). Due to their readily available, noninvasive and information dense nature, 12-lead ECGs are first-line diagnostic tools for screening/evaluation of potential CVDs. However, accurate ECG analysis is limited in places where ECG expertise is not accessible, exacerbated by the decline and lack of available cardiac electrophysiologists especially in rural areas (Johnson, 2024).

The aforementioned facts calls attention to the need for accessible, accurate, and efficient automation of ECG analysis through deep learning. Deep

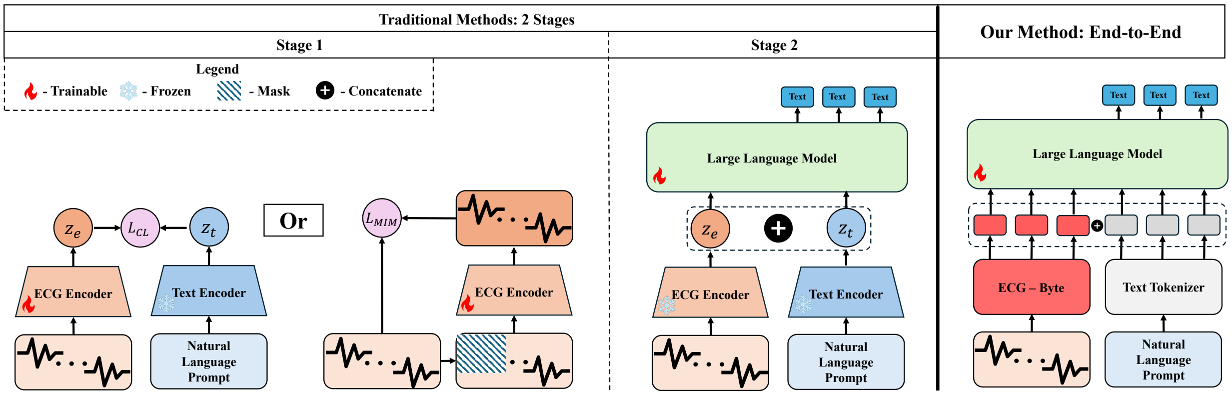

Figure 1: Comparison of traditional methods and our method on ECG language modeling. Traditional methods comprises 2 training stages. The first stage aims to learn a good representation of the 12-lead ECG by training an ECG-specific encoder with a self-supervised learning objective via a combinatorial or individual usage of contrastive learning (L_{CL}) on the hidden states of the ECG z_e and text z_t or masked image modeling (L_{MIM}). The second stage applies the trained encoder to an ECG, which is input alongside a text prompt to a LLM for generation. In contrast, our method is end-to-end and can directly train a LLM for generation utilizing **ECG-Byte**.

learning has reached expert level performance in certain tasks for CVD detection using ECGs (Rajpurkar et al., 2017; Hannun et al., 2019; Qiu et al., 2023b). However, most previous works in this domain have succumbed to a crude classification of hard CVD labels (Choi et al., 2023; Nonaka and Seita, 2020; Martin et al., 2021; Strodthoff et al., 2021). A problem with this approach is that ECGs often do not exclusively fall into one diagnostic category, instead, there may be many soft labels annotated by expert physicians and the accumulation of these soft labels allow a more detailed, nuanced, and clinically useful interpretation of the ECG.

The recent onset of Large Language Models (LLMs) provides an opportunity to take a softer, generative, and consequently more flexible approach to ECG analysis. There are some recent works that have taken advantage of this approach to ECG analysis (Qiu et al., 2023a; Tang et al., 2024b; Wan et al., 2024; Fu et al., 2024; Zhao et al., 2024). A commonality among these works is that they treat multichannel ECGs similarly to images; they first pretrain an encoder specifically for ECGs with some self-supervised learning (SSL) objective, then apply the learned features to finetune a LLM for natural language generation (NLG). However, we observed 2 limitations with this approach: 1) inefficiencies from two-stage training and 2) interpretability challenges with encoder generated features. Pretraining a good ECG specific encoder can be a significant computational burden due to large datasets, model size, and long training times. Additionally, the latent feature vector output by the ECG encoder cannot be mapped back to the signal, making interpretability difficult when utilizing this feature vector for downstream tasks.

In this study, we introduce **ECG-Byte**, an adapted byte pair encoding (BPE) (Gage, 1994) tokenizer designed for end-to-end training for generative ECG language modeling. Inspired by prior works demonstrating the effectiveness of creating discrete tokens from continuous values (Chen et al., 2022; Han et al., 2024), we leverage quantization to represent amplitude ranges as discrete symbols. Using discrete symbols, we obtain string representations of the ECGs to train **ECG-Byte**. We then apply **ECG-Byte** to directly finetune a LLM for conditional autoregressive NLG, where the text output is conditioned on the text prompt and tokenized signal. We found that our end-to-end training approach is competitive in conditional NLG with only half the time and ∼48% of the data required by 2-stage pretraining approaches. Additionally, since the encoded ECG can be reverse transformed back to the original signal, we can interpret token-level attention-based visualizations, whereas it would be impossible to reverse the hidden latent feature vector outputs by a pretrained encoder.

Our contributions are summarized as the following:

1. We present **ECG-Byte**, an adapted BPE tokenizer for end-to-end training on autoregressive, conditional NLG.

2. We empirically show the efficiency of our method and present competitive performance compared to conventional 2-stage pretraining approaches, proposing a new paradigm for conditional NLG with ECGs.

3. We conduct an interpretability study on both **ECG-Byte** and the LLM by respectively observing how **ECG-Byte** is merging ECG signals and by utilizing attention visualizations to observe how the LLM is processing ECGs and text.

2. Related Works

2.1. Deep learning for ECGs

There has been a plethora of works utilizing deep learning for processing ECGs for classification (Rajpurkar et al., 2017; Hannun et al., 2019; Choi et al., 2023; Nonaka and Seita, 2020; Martin et al., 2021; Strodthoff et al., 2021). Most of these works utilize either convolutional neural networks (CNNs) (Rajpurkar et al., 2017) or transformers (Choi et al., 2023) and exhibit excellent performance at classification tasks. There have also been some efforts to frame classification as a retrieval task in order to recover cases similar to the given ECG.(Tang et al., 2024b; Qiu et al., 2023b). However, the retrieval approach may struggle with rare or unique ECG patterns that lack good matches in the existing database, potentially leading to missed or inaccurate diagnoses for unusual cases. Additionally, both classification and retrieval tasks may be crude formulations of processing ECGs, since ECGs typically exhibit many characteristics of overlapping CVDs.

2.2. Large Language Models for ECGs

Generative Large Language Models (LLMs) have given the opportunity to take a softer and more clinically similar approach in processing ECGs by generating physician-vetted clinical statements (Qiu et al., 2023a; Tang et al., 2024b; Wan et al., 2024; Fu et al., 2024; Zhao et al., 2024). The representation of ECG data has been largely considered in previous works which feed the raw signal into an encoder to obtain a latent representation of the ECG data, which then serves as input into a LLM. (Zhao et al., 2024; Wan et al., 2024; Tang et al., 2024b; Choi et al., 2023). In order to get good latent representations of the ECGs, an encoder is first pretrained on a self-supervised learning (SSL) objective (e.g., contrastive learning, masked language/image modeling). Although model performance in terms of label classification has been excellent when using these approaches, we want to be able to generate soft labels akin to clinical notation since ECGs often have overlapping and non-mutually exclusive descriptors. There have been some efforts in this direction (Qiu et al., 2023a; Tang et al., 2024b; Wan et al., 2024; Fu et al., 2024; Zhao et al., 2024), however, they suffer in efficiency (i.e., requires 2 stages of training). In our work, we challenge this 2-stage pretraining approach by transforming the ECG into tokens using **ECG-Byte** and directly training a LLM for NLG.

2.3. Byte Pair Encoding for Domains Outside of Language

The Byte Pair Encoding (BPE) algorithm was first introduced by Gage (1994) for data compression. It was later adapted to the natural language processing (NLP) domain (Sennrich et al., 2016) and has been the favored approach to tokenization for the most popular language models (Grattafiori et al., 2024; Brown et al., 2020) due to its efficiency and robustness to rare words. Byte pair encoding has seen success in representing modalities outside of language, including molecular graphs (Shen and Póczos, 2024), electroencephalogram (EEG) (Klymenko et al., 2023), and more generally, physiological signals.(Tavabi and Lerman, 2021). However, in theses cases the byte pair encoded representations are simply used for classification. Most recently, Tahery et al. (2024) leveraged quantization and BPE to compress ECG signals and pass as inputs to a BERT (Devlin et al., 2019) model for SSL. However, in their work, they only use this representation for classification. As previously mentioned, we believe classification alone may limit aspects of ECG interpretation, thus we utilize these representations for generative diagnosis. Additionally, previous works utilized a pre-existing BPE tokenizer based on SentencePiece (Kudo and Richardson, 2018), and do not conduct further analysis of *how* the BPE algorithm is merging the ECGs.

3. Methods

This section provides detailed information on the datasets, preprocessing, ECG signal encoding with **ECG-Byte**, and LLM training for NLG.

3.1. Dataset and Preprocessing

Dataset In this study, we use variants of the MIMIC-IV ECG (Gow et al., 2023) and PTB-XL datasets (Wagner et al., 2020) for NLG. We use MIMIC-IV ECG pretraining curated by Zhao et al. (2024) that contains question prompts generated by GPT-4o alongside the ECG and clinical notes. Additionally, we use the ECG-QA dataset (Oh et al., 2023), a dataset that uses the ChatGPT API to generate naturalistic, clinically relevant question and answer pairs about the ECG signals from the MIMIC-IV ECG and PTB-XL datasets. The baselines we compare our results with all utilize the *single-verify*, *single-choose*, and *single-query* categorized questions from the ECG-QA dataset. *single-verify* corresponds to yes or no questions, *single-choose* corresponds to where a selected answer is made from two given options, and *single-query* consists of open-ended questions. The ECG signals collected from both both datasets (i.e., MIMIC-IV ECG and PTB-XL) are sampled at 500 Hz for 10 seconds, resulting in a 5000 length, 12 lead ECG.

Preprocessing We preprocess all datasets used in the study in the same manner to maintain consistency. We first convert the ordering of the leads for the MIMIC-IV ECG dataset (i.e., ['I', 'II', 'III', 'aVR', 'aVF', 'aVL', 'V1', 'V2', 'V3', 'V4', 'V5', 'V6'])to the PTB-XL dataset format (i.e., ['I', 'II', 'III', 'aVL', 'aVR', 'aVF', 'V1', 'V2', 'V3', 'V4', 'V5', 'V6']). We then use a notch filter at 50 Hz and 60 Hz to remove powerline interference. Each frequency is targeted with a quality factor of 30 to minimize distortion, and filtering is applied bidirectionally to prevent phase shifts. Next, a fourth-order Butterworth bandpass filter with a range of 0.5–100 Hz isolates relevant ECG components while attenuating high-frequency noise and low-frequency drift. To address baseline wander caused by respiratory or movement artifacts, we bidirectionally apply a fourth-order Butterworth highpass filter with a cutoff of 0.05 Hz. After filtering, we apply wavelet denoising to further reduce noise. Using the Daubechies-6 (db6) wavelet at level 4, we decompose each ECG signal into wavelet coefficients. A soft threshold, based on the median absolute deviation of the detail coefficients, is applied to each coefficient level to suppress noise, ensuring values near zero are excluded from reconstruction. Since 250 Hz is a generally accepted sampling frequency adequate for heartbeat analysis (Kwon et al., 2018), we downsample the 500 Hz sampling frequency to 250. We then segment the 10 second signal to non-overlapping windows of 2 seconds, and use each 12 lead 2 second segment of the ECG signal as input to the model. However, for training the tokenizer, we did not want to introduce this discontinuity across the full 10 seconds. Thus, we utilize the unsegmented, 10 second ECG signal for training **ECG-Byte**. Lastly, during the unsegmented preprocessing pipeline, we record the global 1st and 99th percentiles out of 300,000 samples to utilize in our later steps of training **ECG-Byte** for normalization.

3.2. ECG as Bytes

Sampling Following established practices in NLP (Dagan et al., 2024), we train **ECG-Byte** on a representative subset of the total dataset, selected using stratified sampling based on morphological clustering. To extract features from each unsegmented ECG, we compute statistical measures, frequency and time domain features, morphological characteristics, and wavelet coefficients. Principal Component Analysis (PCA) (Wold et al., 1987) is applied for dimensionality reduction, retaining 95% of the variance, followed by feature scaling. The optimal number of clusters is determined using the Elbow Method and Silhouette Analysis (Rousseeuw, 1987), with the smaller result chosen. K-means clustering (MacQueen, 1967) is then applied to the scaled PCA-transformed features. If K-means fails to yield distinct clusters, DBSCAN (Ester et al., 1996) is used as a fallback. Stratified sampling is performed by randomly selecting ECGs from each cluster in proportion to its size, resulting in a total sample of 200,000 ECGs.

Quantization To ensure consistency across ECG signals, we normalize each input by scaling it to a fixed range and encoding it into a symbolic representation. Let $X \in \mathbb{R}^{C \times T}$ denote an ECG signal matrix, where C is the number of ECG leads and T represents the number of sampled time points per lead. In this study, $C = 12$ and $T = 500$ unless specified otherwise. Let p_1 and p_{99} represent the 1st and 99th percentiles of X across all leads and time points sampled earlier during preprocessing, respectively. The normalization process is defined as follows:

$$X_{\text{norm}} = \frac{X - (p_1 - \epsilon_1)}{(p_{99} + \epsilon_1) - (p_1 - \epsilon_1) + \epsilon_2} \tag{1}$$

where $\epsilon_1 = 0.5$ is a constant to make up for the sampled percentiles and $\epsilon_2 = 10^{-6}$ is a small constant added to prevent division by zero. This transformation shifts and scales X so that the normalized values fall within the range [0, 1]. We then apply clipping to ensure that values remain strictly within this range:

$$X_{\text{clipped}} = \text{clip}(X_{\text{norm}}, 0, 1) \tag{2}$$

Inspired by previous works (Klymenko et al., 2023; Tavabi and Lerman, 2021; Chen et al., 2022; Han et al., 2024), we quantize X_{clipped} into discrete levels for symbolic representation. Let $\mathcal{A}$ be the set of 26 symbols, corresponding to the lowercased letters in the English alphabet, $\mathcal{A} = \{\text{a}, \text{b}, \dots, \text{z}\}$. The alphabet size $|\mathcal{A}| = 26$ defines the number of discrete levels. We scale and floor X_{clipped} to integer values, then take the minimum between the floored value and the maximum number of bins as the following:

$$X_{\text{quant}} = \min(\lfloor X_{\text{clipped}} \times |\mathcal{A}| \rfloor, (|\mathcal{A}| - 1)) \tag{3}$$

Finally, each integer value in X_{quant} is mapped to a corresponding symbol in $\mathcal{A}$ to yield the symbolic signal, which serves as a discrete representation of the ECG. After transforming each ECG signal instance into its symbolic form, we first flatten each symbolic ECG instance $X^{(i)}_{\text{quant}}$ into a 1-dimensional sequence of symbols:

$$X^{(i)}_{\text{symb}} = \text{flatten}(X^{(i)}_{\text{quant}}), \quad X^{(i)}_{\text{symb}} \in \mathcal{A}^{CT} \tag{4}$$

where i indexes over all instances in the dataset, and $X^{(i)}_{\text{symb}}$ is the flattened sequence of symbols of length $C \cdot T$. Next, we concatenate all flattened instances $X^{(1)}_{\text{symb}}, X^{(2)}_{\text{symb}}, \dots, X^{(N)}_{\text{symb}}$ across the entire dataset to form a single, long symbolic sequence:

$$\mathbf{X}_{\text{concat}} = X^{(1)}_{\text{symb}} \| X^{(2)}_{\text{symb}} \| \cdots \| X^{(N)}_{\text{symb}}, \quad \mathbf{X}_{\text{concat}} \in \mathcal{A}^{NCT} \tag{5}$$

where $\|$ denotes the concatenation operation, and N is the total number of instances in the dataset. The concatenated symbolic sequence $\mathbf{X}_{\text{concat}}$ of length $N \cdot C \cdot T$ is then used to train **ECG-Byte**.

ECG-Byte Training Process After obtaining the string representation $\mathbf{X}_{\text{concat}}$ of the ECG dataset, we train **ECG-Byte** to compress the discretized ECG signals by iteratively merging the most frequent byte pairs into single tokens, following the BPE algorithm. The process starts by converting $\mathbf{X}_{\text{concat}}$ into an ID vector of 8-bit unsigned integers and initializing a vocabulary map (`vocab`) for string representations of bytes and a `vocab_tokens` map to encode bytes as singleton lists. IDs and `vocab` are initialized to cover the full byte range (0–255), mapping symbols in $\mathcal{A}$ to ASCII values (97–122), while reserving other byte values for unknown bytes. As merging proceeds, new tokens are assigned unique integer IDs starting from 256, acting as abstract labels for progressively larger token units. For each merge iteration, **ECG-Byte** calculates adjacent byte pair frequencies using a parallelized `get_stats` function, efficiently aggregating counts via a fold-and-reduce strategy. The most frequent pair is identified as the "best pair" to merge, and the `merge` function replaces occurrences of this pair in the ID vector with a new token ID, extending the vocabulary and updating `vocab_tokens` accordingly. This process repeats until the specified number of merges is reached or no pairs remain. The output includes the encoded ID vector, the extended vocabulary map, and a history of merge operations. Existing tokenizers, such as SentencePiece (Kudo and Richardson, 2018) or HuggingFace (Wolf et al., 2020), were not used due to their complexity and integration issues, which hindered interpretability. **ECG-Byte**, implemented in Rust for speed, provides a lightweight, flexible framework for representing ECG signals as discrete tokens while drawing inspiration from HuggingFace's tokenizer (Wolf et al., 2020). Detailed pseudocode for the main training pipeline is provided in Algorithm 1, with `merge` and `get_stats` functions detailed in Appendix A.1.

ECG-Byte Encoding Process After training **ECG-Byte**, we encode any quantized ECG signal X_{symb} by first converting each byte in the ECG signal to a 32-bit unsigned integer and building a trie structure, where each node represents a byte or a merged token sequence from prior encoding steps. The trie is initialized with single-byte tokens (0-255) and is extended with custom token sequences from the learned merge history. For each byte sequence in the input, the encoding function traverses the trie to find the longest match, replacing matched sequences with their assigned token IDs. If no match is found, the byte is added to the output as-is. The final encoded sequence is returned as `output_ids`, where we will denote as X_{ID}. We provide the detailed pseudocode of the encoding process in Algorithm 2.

Algorithm 1: Training Process for **ECG-Byte**

```
Input: Input X_concat, Number of merges num_merges
Output: Tuple containing final encoded IDs, vocabulary map, and merge history
Function byte_pair_encoding(X_concat, num_merges):
    Convert X_concat to ID vector ids by casting each byte to u32;
    Initialize vocab with mappings from IDs 0 to 255 to their string representations;
    Initialize vocab_tokens with mappings from IDs 0 to 255 to singleton lists;
    Initialize empty list merges;
    for i ← 0 to num_merges exclusive do
        pairs ← get_stats(ids) using parallel processing;
        if pairs is empty then
            break
        end
        best_pair ← Pair in pairs with highest frequency;
        if best_pair is not found then
            break
        end
        new_id ← 256 + i;
        ids ← merge(ids, best_pair, new_id);
        vocab[new_id] ← concat(vocab[best_pair.0], vocab[best_pair.1]);
        new_token ← concat(vocab_tokens[best_pair.0], vocab_tokens[best_pair.1]);
        vocab_tokens[new_id] ← new_token;
        merges.append(new_token, new_id);
    end
    return (ids, vocab, merges);
```

3.3. Large Language Model

In this study, we utilize the Llama-3.2-1B (Grattafiori et al., 2024) checkpoint through the HuggingFace API (Wolf et al., 2020) unless specified otherwise. The Llama 3.2 series model is a variant of the Llama 3 models (Grattafiori et al., 2024) and support context

Algorithm 2: Encoding Process for **ECG-Byte**

```
Input: Input X_symb, Merge history merges containing pairs of token sequences and their token IDs
Output: Vector of encoded IDs
Function encode(X_symb, merges):
    ids ← Convert X_symb to vector of u32 by casting each byte;
    Initialize root TrieNode trie_root using TrieNode::new();
    for b ← 0 to 255 do
        insert(trie_root, [b], b);
    end
    foreach (token_sequence, token_id) in merges do
        insert(trie_root, token_sequence, token_id);
    end
    Initialize empty list output_ids;
    i ← 0;
    while i < length of ids do
        node ← trie_root;
        match_len ← 0;
        match_id ← None;
        for j ← i to length of ids do
            id ← ids[j];
            if id exists in node.children then
                node ← node.children[id];
                if node.token_id is not None then
                    match_len ← j - i + 1;
                    match_id ← node.token_id;
                end
            end
            else
                break
            end
        end
        if match_id is not None then
            output_ids.append(match_id);
            i ← i + match_len;
        end
        else
            output_ids.append(ids[i]);
            i ← i + 1;
        end
    end
    return output_ids;
```

lengths of up to 128k tokens. They are notable for their superior performance despite having 1 billion parameters, making them highly efficient and capable models to test our methodology upon. We also provide an ablation study in subsection 5.4 where we utilize other popular LLMs, such as GPT2 XL 1.5B (Radford et al., 2019), Gemma 2B (Team et al., 2024), and OPT 1.3B (Zhang et al., 2022).

3.4. Learning Objective

The learning objective for training the LLM considers a sequence composed of three parts, $\{X_{\text{ID}}, Q, \mathcal{S}\}$, where $X_{\text{ID}} \in \mathcal{V}^M$ represents the encoded ECG sequence of length $l_{X_{\text{ID}}} = |X_{\text{ID}}|$, with each token drawn from the extended vocabulary $\mathcal{V}$ of size M, Q represents the tokenized question, and $\mathcal{S}$ denotes the tokenized answer sequence. The input sequence includes special tokens: - `[BOS]` as the beginning-of-sequence token, - `[SIG_START]` and `[SIG_END]` to indicate the start and end of the encoded ECG sequence, and - `[EOS]` as the end-of-sequence token for the generated answer. The motivation for adding `[SIG_START]` and `[SIG_END]` special tokens is inspired by Liu et al. (2023), where they utilize special tokens indicating the start and end of the image. Thus, the full input sequence is structured as:

$$\texttt{[BOS]} \,\|\, \texttt{[SIG\_START]} \,\|\, X_{\text{ID}} \,\|\, \texttt{[SIG\_END]} \,\|\, Q \| \mathcal{S} \| \, \texttt{[EOS]}\,,$$

where $\|$ denotes concatenation. Let $l_Q = |Q|$, $l_{\mathcal{S}} = |\mathcal{S}|$, and L be the total sequence length, given by:

$$L = 1 + 1 + l_{X_{\text{ID}}} + 1 + l_Q + l_{\mathcal{S}} + 1,$$

accounting for the `[BOS]`, `[SIG_START]`, `[SIG_END]`, and `[EOS]` tokens. The autoregressive objective maximizes the likelihood of each token in $\mathcal{S}\|\texttt{[EOS]}$ conditioned on the preceding context $\texttt{Context} = \{\texttt{[BOS]}, \texttt{[SIG\_START]}, X_{\text{ID}}, \texttt{[SIG\_END]}, Q\}$ and the previous tokens in $\mathcal{S}$. The objective is formulated as follows:

$$\mathcal{L}_{\text{NLL}} = - \sum_{l^{\text{'}} = l_{X_{\text{ID}}} + l_Q + 4}^{L} \log P(s_{l^{\text{'}}} \mid \texttt{Context}, s_{<l^{\text{'}}}; \theta), \tag{6}$$

where $s_{l^{\text{'}}} = \mathcal{S}_{l^{\text{'}} - (l_{X_{\text{ID}}} + l_Q + 4)}$ is the $(l' - (l_{X_{\text{ID}}} + l_Q + 4))$-th token in $\mathcal{S}\|\texttt{[EOS]}$, and $s_{<l^{\text{'}}} = \{s_1, s_2, \ldots, s_{l^{\text{'}} - (l_{X_{\text{ID}}} + l_Q + 4) - 1}\}$ denotes all tokens in $\mathcal{S}\|\texttt{[EOS]}$ preceding $s_{l^{\text{'}}}$.

4. Experiments

4.1. Experimental Settings

We fine-tuned the LLM using the AdamW optimizer (Kingma and Ba, 2017) with a learning rate of $1e-4$, weight decay of $1e-2$, and a custom learning rate scheduler. This scheduler applies an initial learning rate init_lr scaled by the model's hidden dimension $(\text{d}_{\text{model}}{}^{-0.5})$ and dynamically adjusts it based on training steps, with a warm-up phase of 500 steps. The learning rate at step n_{steps} is updated as $\text{lr} = \text{init\_lr} \times \min\left(\text{n}_{\text{steps}}{}^{-0.5},\ \text{n}_{\text{warmup}}{}^{-1.5} \times \text{n}_{\text{steps}}\right)$. We set $\beta_1 = 0.9$, $\beta_2 = 0.99$, $\epsilon = 1e-8$, batch size 2, and trained for 1 epoch. Additionally, we only train on a randomly sampled subset of 400,000 ECG instances for each respective dataset due to computational resources, unless specified otherwise. We also utilize LoRA (Hu et al., 2021) to finetune the LLM with rank = 16, $\alpha_{LoRA} = 32$, and dropout = 0.05. We conduct our experiments on 4 NVIDIA RTX A6000 48 GB GPUS.

During inference, we evaluate our model with number of merges `num_merges` = 3500, sequence length $L = 1024$, and ECG length $T = 500$ unless specified otherwise. We use popular metrics for NLG namely the BLEU-4 (Papineni et al., 2002), Rouge-L (Lin, 2004), Meteor (Banerjee and Lavie, 2005), and BertScore F1 (Zhang et al., 2020) metrics.

5. Results

5.1. Natural Language Generation

We present our main results in Table 1, comparing **ECG-Byte** with prior works and self-implemented two-stage pretraining methods. Notably, Zhao et al. (2024) is not directly comparable due to differing data splits and pretraining datasets, though reported metrics use the same datasets. Zhao et al. (2024) train on the *full* MIMIC-IV ECG Pretrain dataset, finetune on an instruction-tuning dataset for ECG-related conversations, and evaluate on PTB-XL (Wagner et al., 2020) using a unified question: "Could you please help me explain my ECG?" To establish baselines, we implement generic two-stage pretraining methods: L_{CL}, L_{MIM}, and $L_{CL} + L_{MIM}$. Here, L_{CL} employs contrastive learning (Liu et al., 2024; Gopal et al., 2021; Pham et al., 2024; Kiyasseh et al., 2021), L_{MIM} uses Masked Image Modeling (MIM) (Choi et al., 2023; Na et al., 2024; Yang et al., 2022), and $L_{CL} + L_{MIM}$ combines both (Oh et al., 2022;

Table 1: NLG mean results with standard deviations over 5 random seeds comparing against different baselines.

Method	Trained Dataset	Inferenced Dataset	BLEU-4	Rouge-L	Meteor	BertScore F1
ECG-Chat (Zhao et al., 2024)	MIMIC-IV ECG Pretrain	PTB-XL	**11.19**	29.93	**35.10**	-
L_{CL}			8.10 ± 0.25	31.36 ± 0.31	27.55 ± 0.36	89.35 ± 0.04
L_{MIM}			6.21 ± 0.22	30.63 ± 0.13	24.91 ± 0.14	**90.44** ± 0.04
L_{MERL} (Liu et al., 2024)			10.22 ± 0.25	32.95 ± 0.12	25.60 ± 0.17	89.94 ± 0.01
$L_{CL} + L_{MIM}$			9.33 ± 0.22	30.45 ± 0.21	24.37 ± 0.36	90.29 ± 0.02
ECG-Byte			11.00 ± 0.19	**33.41** ± 0.05	24.95 ± 0.09	90.02 ± 0.01
L_{CL}	ECG-QA MIMIC-IV	ECG-QA MIMIC-IV	10.22 ± 0.06	38.41 ± 0.48	24.66 ± 0.23	90.42 ± 0.09
L_{MIM}			7.90 ± 0.23	29.28 ± 0.38	19.03 ± 0.11	67.91 ± 0.17
L_{MERL} (Liu et al., 2024)			10.95 ± 0.24	38.18 ± 0.58	26.24 ± 0.36	90.80 ± 0.06
$L_{CL} + L_{MIM}$			8.57 ± 0.14	34.00 ± 0.25	25.22 ± 0.30	87.72 ± 0.04
ECG-Byte			**11.23** ± 0.12	**42.49** ± 0.53	**27.08** ± 0.15	**91.30** ± 0.04
L_{CL}	ECG-QA PTB-XL	ECG-QA PTB-XL	8.89 ± 0.25	28.63 ± 0.47	18.45 ± 0.31	72.63 ± 0.40
L_{MIM}			**15.14** ± 0.28	46.71 ± 0.41	**29.64** ± 0.30	92.12 ± 0.10
L_{MERL} (Liu et al., 2024)			13.84 ± 0.19	40.14 ± 0.39	26.24 ± 0.35	91.88 ± 0.09
$L_{CL} + L_{MIM}$			14.72 ± 0.27	42.88 ± 0.13	28.25 ± 0.27	89.40 ± 0.01
ECG-Byte			13.93 ± 0.21	**47.08** ± 0.56	29.17 ± 0.31	**92.53** ± 0.07

McKeen et al., 2024). These approaches utilize pretrained CLIP (Radford et al., 2021) and ViT (Dosovitskiy et al., 2021), where ECG signals are transformed into three-channel images for finetuning. We adapt Liu et al. (2024)'s state-of-the-art contrastive method (L_{MERL}) for fair comparison. Their most effective model uses a 1D ResNet backbone (He et al., 2015); hence, we employ ResNet101 for direct ECG signal processing. For L_{CL}, L_{MIM}, $L_{CL} + L_{MIM}$, and L_{MERL}, training is conducted on the *full*, preprocessed MIMIC-IV ECG dataset with a batch size of 64 during the first stage. Implementation details for both training stages are in Appendix B. Table 1 demonstrates **ECG-Byte**'s effectiveness, showing competitive or superior performance across all metrics and datasets compared to other methods. Qualitative examples are provided in Appendix C.2.

5.2. Cross Dataset Transferability

We present the results of cross-dataset transferability in Table 2, comparing our approach, **ECG-Byte**, with two-stage pretraining methods. **ECG-Byte** achieves the best zero-shot transfer performance from the ECG-QA PTB-XL dataset to the ECG-QA MIMIC-IV dataset. When transferring from the ECG-QA MIMIC-IV dataset to the ECG-QA PTB-XL dataset, although other 2-stage pretraining methods demonstrate higher performance, **ECG-Byte** maintains competitive results across all metrics.

5.3. Efficiency of ECG-Byte

We compare the efficiency of our end-to-end approach using **ECG-Byte** with 2-stage pretraining methods in Table 3. First, we examine the amount of data required for each method. As previously noted, the first stage of the two-stage pretraining methods is trained on the full MIMIC-IV ECG dataset (Johnson et al., 2023) using segmented ECGs, which are also used as input during the second stage. While **ECG-Byte** is trained on unsegmented ECGs, we convert the number of unsegmented ECGs to an equivalent count of segmented ECGs. Additionally, the reduced data requirement for **ECG-Byte** is due to our sampling approach, where only a subset of ECGs is used to train the tokenizer. Under these settings, our proposed method achieves competitive results using approximately ∼48% of the data required for 2-stage pretraining methods. In terms of training time, our approach requires less than *half* the time needed for two-stage pretraining. The training time for the two-stage methods is averaged across our self-implemented approaches (L_{CL}, L_{MIM}, and $L_{CL} + L_{MIM}$) and the L_{MERL} method proposed by Liu et al. (2024).

5.4. Ablation Study

We conduct several ablation studies to show the variability of performance with **ECG-Byte** when we alter the LLM used for finetuning, use different sequence lengths L when inputting to the LLM, training **ECG-Byte** with various number of merges

Table 2: Mean results with standard deviations over 5 random seeds on zero shot cross-dataset transferability.

Method	Trained Dataset	Inferenced Dataset	BLEU-4	Rouge-L	Meteor	BertScore F1
L_{CL}			11.64 ± 0.45	41.48 ± 0.11	25.74 ± 0.13	91.24 ± 0.05
L_{MIM}			**11.70** ± 0.29	**42.22** ± 0.28	**26.41** ± 0.10	91.51 ± 0.03
L_{MERL} (Liu et al., 2024)	ECG-QA MIMIC-IV	ECG-QA PTB-XL	11.53 ± 0.19	39.23 ± 0.40	25.58 ± 0.28	**91.59** ± 0.03
$L_{CL} + L_{MIM}$			9.71 ± 0.10	35.10 ± 0.28	24.91 ± 0.19	87.88 ± 0.08
ECG-Byte			8.70 ± 0.04	40.39 ± 0.40	23.29 ± 0.18	91.51 ± 0.03
L_{CL}			5.10 ± 0.04	22.77 ± 0.28	14.63 ± 0.32	77.89 ± 0.13
L_{MIM}			7.68 ± 0.46	**35.77** ± 0.13	**22.32** ± 0.33	90.28 ± 0.07
L_{MERL} (Liu et al., 2024)	ECG-QA PTB-XL	ECG-QA MIMIC-IV	7.39 ± 0.15	28.33 ± 0.58	18.59 ± 0.35	89.30 ± 0.05
$L_{CL} + L_{MIM}$			7.49 ± 0.21	30.53 ± 0.59	20.25 ± 0.27	86.53 ± 0.11
ECG-Byte			**7.86** ± 0.13	35.01 ± 0.41	21.49 ± 0.24	**90.29** ± 0.07

Table 3: Efficiency of our method compared against 2-stage pretraining methods.

	1st Stage	2nd Stage	ECG-Byte	end-to-end
# of Data	2,513,435	400,000	1,000,000	400,000
Total # of Data	2,913,435		**1,400,000**	
Time (minutes)	~1258.50	~469.25	~385.12	~420.32
Total Time (minutes)	~1727.75		**~805.44**	

`num_merges`, and varying ECG lengths T. With the exception of the ablating parameter, we fix all other parameters to `num_merges` = 3500, $L = 1024$, and $T = 500$. We report results on the test set of the PTB-XL variant of ECG-QA (Oh et al., 2023) unless specified otherwise.

Table 4: Ablation study on using different LLMs.

LLM	BLEU-4	Rouge-L	Meteor	BertScore F1
GPT2 XL 1.5B (Radford et al., 2019)	12.30 ± 0.19	41.33 ± 0.57	26.48 ± 0.33	92.00 ± 0.06
Gemma 2B (Team et al., 2024)	13.78 ± 0.18	45.48 ± 0.55	28.32 ± 0.23	92.01 ± 0.02
OPT 1.3B (Zhang et al., 2022)	12.26 ± 0.20	41.84 ± 0.52	26.21 ± 0.29	91.78 ± 0.04
Llama 3.2 1B (Grattafiori et al., 2024)	**13.93** ± 0.21	**47.08** ± 0.56	**29.17** ± 0.31	**92.53** ± 0.07

Different LLMs We show the variability in performance of **ECG-Byte** when using different LLMs with similar numbers of parameters in Table 4. While Llama 3.2 1B (Grattafiori et al., 2024) achieves the best results, GPT2 XL 1.5B (Radford et al., 2019), Gemma 2B (Team et al., 2024), and OPT 1.3B (Zhang et al., 2022) also deliver comparable performances. These findings demonstrate that our method is not limited to Llama 3.2 1B but can achieve similar results across a variety of LLMs.

Sequence Length Input lengths for LLMs are an important parameter to consider for efficient training since the calculation of attention is quadratic with respect to the input length (Vaswani et al., 2023). We present results on different sequence lengths L in Table 5. Although the difference is not substantial, we can see that when $L = 1024$ and $L = 2048$ the model yields higher performance then $L = 512$. We attribute this to the rate of truncation and padding for the encoded ECG. Observing Figure 2, most ECGs were being encoded to token sequence lengths of around 500 to 1500. Therefore, we hypothesize that when $L = 512$ a large portion of the ECG token sequence gets truncated, resulting in lower performance.

Table 5: Ablation study on varying sequence lengths L.

L	BLEU-4	Rouge-L	Meteor	BertScore F1
512	13.61 ± 0.15	**48.15** ± 0.57	29.10 ± 0.28	92.41 ± 0.05
1024	**13.93** ± 0.21	47.08 ± 0.56	**29.17** ± 0.31	**92.53** ± 0.07
2048	13.88 ± 0.22	45.21 ± 0.48	28.31 ± 0.27	90.88 ± 0.02

Number of Merges The number of merges `num_merges` performed during training **ECG-Byte** corresponds to how much the algorithm compresses the concatenated sequence of quantized ECGS $\mathbf{X}_{\text{concat}}$. More `num_merges` means more compression, which can affect the expressiveness of the encoded sequence. In Table 6, we show the performance of our method with different `num_merges`. The results indicate that while performance varies slightly with the number of merges, values of `num_merges` greater than 500 generally yield similar outcomes.

ECG length Lastly, we show the effect of the length T being considered when encoding the ECG

Table 6: Ablation study on varying number of merges num_merges.

num_merges	BLEU-4	Rouge-L	Meteor	BertScore F1
500	13.61 ± 0.53	46.50 ± 0.28	28.49 ± 0.49	92.33 ± 0.02
1750	14.50 ± 0.25	46.74 ± 0.48	30.03 ± 0.25	**92.55** ± 0.01
2500	**15.10** ± 0.39	46.37 ± 0.28	**30.12** ± 0.23	92.53 ± 0.05
3500	13.93 ± 0.21	**47.08** ± 0.56	29.17 ± 0.31	92.53 ± 0.07

with **ECG-Byte** in Table 7. We want to note that for the results of $T = 2000$, the full unsegmented ECG is utilized. Consequently, the number of instances available is less than the targeted dataset size of 400,000 (i.e., 97,244). Thus, when $T = 2000$, we use the full dataset to train the model. For shorter segment lengths, such as $T = 250$ and $T = 500$, the model demonstrates strong performances indicating that shorter segments can effectively preserve relevant information for NLG. Interestingly, for $T = 2000$, the model achieves the highest performance across all metrics. This suggests that when the model is trained with the full 10 second encoded ECG, it benefits from richer contextual information present in the complete ECG waveform.

Table 7: Ablation study on varying lengths T.

T	BLEU-4	Rouge-L	Meteor	BertScore F1
250	12.64 ± 0.20	47.31 ± 0.26	27.97 ± 0.21	92.32 ± 0.06
500	13.93 ± 0.21	47.08 ± 0.56	29.17 ± 0.31	92.53 ± 0.07
1250	11.01 ± 0.19	43.84 ± 0.28	25.49 ± 0.20	**93.07** ± 0.03
2000	**14.54** ± 0.17	**48.03** ± 0.27	**32.11** ± 0.22	92.91 ± 0.04

5.5. ECG-Byte Analysis

We analyze **ECG-Byte** by visualizing the usage of merged tokens, length of the encoded ECG, and mapping between th encoded tokens and original ECG. Unless specified otherwise, we analyze **ECG-Byte** when num_merges $= 3500$, $L = 1024$, and $T = 500$.

Token Usage and Length Distribution We examine the token usage and length distributions for **ECG-Byte** with num_merges $= 3500$ on a subsample of 277,840 ECGs from the PTB-XL dataset. The left panel of Figure 2 displays the token usage distribution, showing token frequency (y-axis) ranked in descending order (x-axis). A small subset of tokens dominates the occurrences, while the rest are infrequently used—a typical characteristic of BPE-based tokenization, where common patterns are compressed into frequent tokens and rare patterns into infrequent ones. The right panel of Figure 2 illustrates the token length distribution of the encoded ECGs, with most falling between 500 and 1000 tokens, demonstrating **ECG-Byte**'s effective compression of the original signal. Additional examples of these distributions are provided in Appendix A.3.

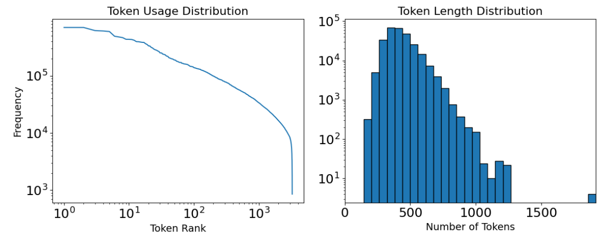

Figure 2: Plots of the token usage and length distributions for **ECG-Byte** where num_merges $=$ 3500. More examples with varying num_merges are provided in Appendix A.3.

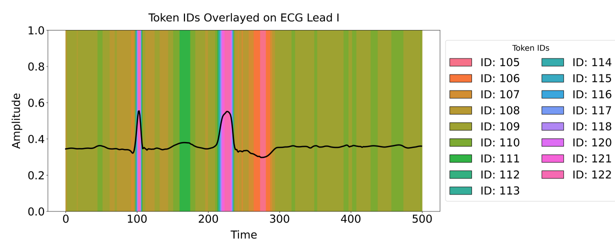

Figure 3: A mapping between tokens used for a given ECG Lead I. More examples are provided in Appendix A.2.

Token to ECG Mapping To illustrate how **ECG-Byte** encodes ECG signals, we analyze the mapping between tokens and signal features. Figure 3 shows an example Lead I ECG signal with unique token IDs (represented by different colors) overlayed. The P wave, QRS complex, and T wave are distinctly captured by different tokens, though this precision varies across instances. As demonstrated, **ECG-Byte** effectively merges key regions of the signal. Additional examples are provided in Appendix A.2 due to page limitations.

Attention Visualizations Figure 4 visualizes attention weights across a selected ECG lead and text portions of the input after training. We focus on one lead due to the uniformity of attention across encoded signal tokens. For interpretability, the reversed ECG signal is overlayed on the encoded ECG. The model primarily attends to the textual portion of the input sequence, as shown in Figure 4. Previous studies have debated whether attention visualizations are inherently explainable (Jain and Wallace, 2019; Wiegreffe and Pinter, 2019) and explored their role in vision-language models (Aflalo et al., 2022; Woo et al., 2024; Arif et al., 2024; Cui et al., 2024). These works often observe minimal attention to visual input, with models relying primarily on text. We hypothesize that a similar phenomenon occurs in Figure 4, as the ECG tokens, though represented like text, are 1) newly introduced and 2) perceived as a different modality (e.g., vision). Additional examples are provided in Appendix A.4.

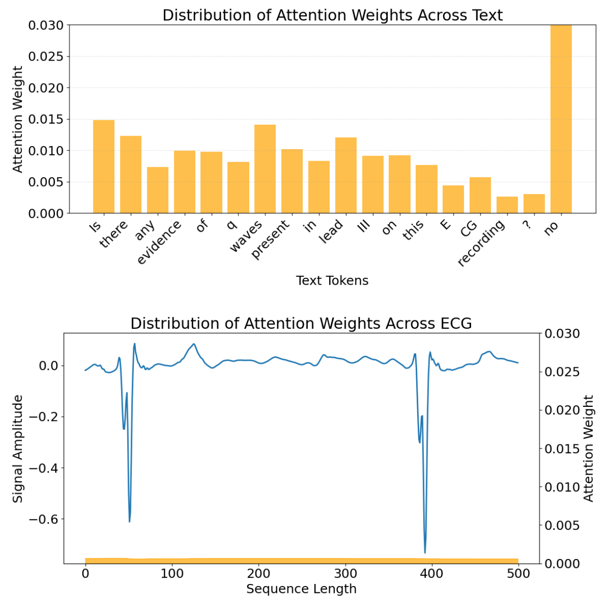

Figure 4: The attention weight overlayed on top of both the text (top) and ECG (bottom). More examples are provided in Appendix A.4.

6. Discussion and Conclusion

In this study, we introduce **ECG-Byte**, a custom BPE algorithm to encode ECGs into a discrete sequence of tokens for conditional autoregressive NLG. **ECG-Byte** introduces a paradigm shift in generative ECG language modeling by enabling efficient end-to-end training, compared to traditional two-stage pretraining approaches. Our pipeline demonstrates strong performances, achieving results comparable to two-stage methods while requiring only *half* the training time and approximately 48% of the data. In addition to its efficiency, **ECG-Byte** enhances interpretability. By analyzing its underlying mechanism, we observe that critical ECG regions, such as the P wave, the QRS complex, and the T wave, are effectively grouped during tokenization, as illustrated in Figure 3. Furthermore, the reversibility of the compressed token sequence allows us to trace each token back to its original ECG signal segment, providing insight into the specific portions of the signal attended to by the model. However, as shown in Figure 4, the model's attention weight distribution resembles that of vision language models, focusing primarily on the textual components of the input sequence during generation.

This work is in its early stages and needs further exploration. Future directions include: (1) refining BPE merging rules to better capture ECG-specific features, (2) adopting more advanced quantization techniques that preserve time-series characteristics (Carson et al., 2024b; Elsworth and Güttel, 2020; Carson et al., 2024a), (3) introducing stronger modality-specific distinctions, such as embeddings beyond `[SIG_START]` and `[SIG_END]` (Gui et al., 2023), and (4) extending **ECG-Byte** for conversational tasks through instruction tuning.

Limitations One of the main limitations of this work is the scale in terms of computing and data. Since we only used a subset of the data to train and test our method, we were unable to train the model to its full potential. However, even with only using a small subset of the data, we are able to see extremely promising results compared to other 2-stage SSL pretraining methods. Therefore, we do not view this limitation as a major bottleneck.

Acknowledgments

This work is done in collaboration with the Mario Lemieux Center for Heart Rhythm Care at Allegheny General Hospital. We thank Wenhao Ding, Haohong Lin, Shiqi Liu, and Hyoeun Kang for the valuable discussions.

References

Estelle Aflalo, Meng Du, Shao-Yen Tseng, Yongfei Liu, Chenfei Wu, Nan Duan, and Vasudev Lal. Vl-interpret: An interactive visualization tool for interpreting vision-language transformers, 2022. URL https://arxiv.org/abs/2203.17247.

Kazi Hasan Ibn Arif, JinYi Yoon, Dimitrios S. Nikolopoulos, Hans Vandierendonck, Deepu John, and Bo Ji. Hired: Attention-guided token dropping for efficient inference of high-resolution vision-language models in resource-constrained environments, 2024. URL https://arxiv.org/abs/2408.10945.

Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *IEEvaluation@ACL*, 2005.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

Erin Carson, Xinye Chen, and Cheng Kang. Llm-abba: Understand time series via symbolic approximation, 2024a. URL https://arxiv.org/abs/2411.18506.

Erin Carson, Xinye Chen, and Cheng Kang. Quantized symbolic time series approximation, 2024b. URL https://arxiv.org/abs/2411.15209.

Ting Chen, Saurabh Saxena, Lala Li, David J. Fleet, and Geoffrey Hinton. Pix2seq: A language modeling framework for object detection, 2022.

Seokmin Choi, Sajad Mousavi, Phillip Si, Haben G. Yhdego, Fatemeh Khadem, and Fatemeh Afghah. Ecgbert: Understanding hidden language of ecgs with self-supervised representation learning, 2023.

Chenhang Cui, Jiabing Yang, Yiyang Zhou, Peng Xia, Ying Wei, and Huaxiu Yao. Fading focus: Mitigating visual attention degradation in large vision-language models, 2024. URL https://openreview.net/forum?id=gam5LiMPKT.

Gautier Dagan, Gabriel Synnaeve, and Baptiste Rozière. Getting the most out of your tokenizer for pre-training and domain adaptation, 2024. URL https://arxiv.org/abs/2402.01035.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.

Steven Elsworth and Stefan Güttel. Abba: Adaptive brownian bridge-based symbolic aggregation of time series, 2020. URL https://arxiv.org/abs/2003.12469.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, page 226–231. AAAI Press, 1996.

Guohua Fu, Jianwei Zheng, Islam Abudayyeh, Chizobam Ani, Cyril Rakovski, Louis Ehwerhemuepha, Hanna Lu, Yongjuan Guo, Shenglin Liu, Huimin Chu, and Bing Yang. Cardiogpt: An ecg interpretation generation model. *IEEE Access*, PP: 1–1, 01 2024. doi: 10.1109/ACCESS.2024.3384349.

Philip Gage. A new algorithm for data compression. *The C Users Journal archive*, 12:23–38, 1994. URL https://api.semanticscholar.org/CorpusID:59804030.

Bryan Gopal, Ryan W. Han, Gautham Raghupathi, Andrew Y. Ng, Geoffrey H. Tison, and Pranav Rajpurkar. 3kg: Contrastive learning of 12-lead electrocardiograms using physiologically-inspired augmentations, 2021. URL https://arxiv.org/abs/2106.04452.

Brian Gow, Tom Pollard, Larry A Nathanson, Alistair Johnson, Benjamin Moody, Chrystinne Fernandes, Nathaniel Greenbaum, Jonathan W Waks, Parastou Eslami, Tanner Carbonati, Ashish Chaudhari, Elizabeth Herbst, Dana Moukheiber, Seth Berkowitz, Roger Mark, and Steven Horng. Mimic-iv-ecg: Diagnostic electrocardiogram matched subset, 2023. URL https://physionet.org/content/mimic-iv-ecg/1.0/.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Liangke Gui, Yingshan Chang, Qiuyuan Huang, Subhojit Som, Alex Hauptmann, Jianfeng Gao, and Yonatan Bisk. Training vision-language transformers from captions, 2023. URL https://arxiv.org/abs/2205.09256.

William Jongwon Han, Diana Gomez, Avi Alok, Chaojing Duan, Michael A. Rosenberg, Douglas Weber, Emerson Liu, and Ding Zhao. Interpretation of intracardiac electrograms through textual representations, 2024. URL https://arxiv.org/abs/2402.01115.

Awni Y. Hannun, Pranav Rajpurkar, Masoumeh Haghpanahi, Geoffrey H. Tison, Codie Bourn, Mintu P. Turakhia, and Andrew Y. Ng. Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network. *Nature Medicine*, 25: 65–69, 01 2019. doi: 10.1038/s41591-018-0268-3. URL https://www.nature.com/articles/s41591-018-0268-3.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. URL https://arxiv.org/abs/1512.03385.

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, 2021. URL https://arxiv.org/abs/2106.09685.

Sarthak Jain and Byron C. Wallace. Attention is not explanation, 2019.

Alistair E. W. Johnson, Lucas Bulgarelli, Lu Shen, Alvin Gayles, Ayad Shammout, Steven Horng, Tom J. Pollard, Benjamin Moody, Brian Gow, Li-wei H. Lehman, Leo A. Celi, and Roger G. Mark. Mimic-iv, a freely accessible electronic health record dataset. *Scientific Data*, 10, 01 2023. doi: 10.1038/s41597-022-01899-x.

Mark Johnson. Counties most in need of cardiologists are the most likely to have none, 07 2024. URL https://www.washingtonpost.com/science/2024/07/29/cardiologists-rural-counties-shortage/.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

Dani Kiyasseh, Tingting Zhu, and David A. Clifton. Clocs: Contrastive learning of cardiac signals across space, time, and patients, 2021. URL https://arxiv.org/abs/2005.13249.

Mykola Klymenko, Sam M Doesburg, George Medvedev, Pengcheng Xi, Urs Ribary, and Vasily A Vakorin. Byte-pair encoding for classifying routine clinical electroencephalograms in adults over the lifespan. *IEEE Journal of Biomedical and Health Informatics*, pages 1–11, 01 2023. doi: 10.1109/jbhi.2023.3236264.

Taku Kudo and John Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing, 2018. URL https://arxiv.org/abs/1808.06226.

Ohhwan Kwon, Jinwoo Jeong, Hyung Bin Kim, In Ho Kwon, Song Yi Park, Ji Eun Kim, and Yuri Choi. Electrocardiogram sampling frequency range acceptable for heart rate variability analysis. *Healthcare Informatics Research*, 24:198, 2018. doi: 10.4258/hir.2018.24.3.198. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6085204/.

Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *ACL 2004*, 2004.

Che Liu, Zhongwei Wan, Cheng Ouyang, Anand Shah, Wenjia Bai, and Rossella Arcucci. Zero-shot ecg classification with multimodal learning and test-time clinical knowledge enhancement, 2024. URL https://arxiv.org/abs/2403.06659.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning, 2023. URL https://arxiv.org/abs/2304.08485.

J MacQueen. Some methods for classification and analysis of multivariate observations. *Project Euclid*, 5.1:281–298, 1967.

Harold Martin, Ulyana Morar, Walter Izquierdo, Mercedes Cabrerizo, Anastasio Cabrera, and Malek Adjouadi. Real-time frequency-independent single-lead and single-beat myocardial infarction detection. *Artificial intelligence in medicine*, 121: 102179, 2021.

Kaden McKeen, Laura Oliva, Sameer Masood, Augustin Toma, Barry Rubin, and Bo Wang. Ecg-fm: An open electrocardiogram foundation model, 2024. URL https://arxiv.org/abs/2408.05178.

Yeongyeon Na, Minje Park, Yunwon Tae, and Sunghoon Joo. Guiding masked representation learning to capture spatio-temporal relationship of electrocardiogram, 2024. URL https://arxiv.org/abs/2402.09450.

Naoki Nonaka and Jun Seita. Electrocardiogram classification by modified efficientnet with data augmentation. In *2020 Computing in Cardiology*, pages 1–4. IEEE, 2020.

Jungwoo Oh, Hyunseung Chung, Joon myoung Kwon, Dong gyun Hong, and Edward Choi. Lead-agnostic self-supervised learning for local and global representations of electrocardiogram, 2022. URL https://arxiv.org/abs/2203.06889.

Jungwoo Oh, Gyubok Lee, Seongsu Bae, Joon myoung Kwon, and Edward Choi. Ecg-qa: A comprehensive question answering dataset combined with electrocardiogram, 2023. URL https://arxiv.org/abs/2306.15681.

World Health Organization. Cardiovascular diseases, 2024. URL https://www.who.int/health-topics/cardiovascular-diseases#tab=tab_1.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL*, 2002.

Manh Pham, Aaqib Saeed, and Dong Ma. C-melt: Contrastive enhanced masked auto-encoders for ecg-language pre-training, 2024. URL https://arxiv.org/abs/2410.02131.

Jielin Qiu, William Han, Jiacheng Zhu, Mengdi Xu, Michael Rosenberg, Emerson Liu, Douglas Weber, and Ding Zhao. Transfer knowledge from natural language to electrocardiography: Can we detect cardiovascular disease through language models? In Andreas Vlachos and Isabelle Augenstein, editors, *Findings of the Association for Computational Linguistics: EACL 2023*, pages 442–453, Dubrovnik, Croatia, May 2023a. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-eacl.33. URL https://aclanthology.org/2023.findings-eacl.33.

Jielin Qiu, Jiacheng Zhu, Shiqi Liu, William Han, Jingqi Zhang, Chaojing Duan, Michael A. Rosenberg, Emerson Liu, Douglas Weber, and Ding Zhao. Automated cardiovascular record retrieval by multimodal learning between electrocardiogram and clinical report. In Stefan Hegselmann, Antonio Parziale, Divya Shanmugam, Shengpu Tang, Mercy Nyamewaa Asiedu, Serina Chang, Tom Hartvigsen, and Harvineet Singh, editors, *Proceedings of the 3rd Machine Learning for Health Symposium*, volume 225 of *Proceedings of Machine Learning Research*, pages 480–497. PMLR, 10 Dec 2023b. URL https://proceedings.mlr.press/v225/qiu23a.html.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners, 2019. URL https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

Pranav Rajpurkar, Awni Y. Hannun, Masoumeh Haghpanahi, Codie Bourn, and Andrew Y. Ng. Cardiologist-level arrhythmia detection with convolutional neural networks, 2017. URL https://arxiv.org/abs/1707.01836.

Peter J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987. ISSN 0377-0427. doi: https://doi.org/10.1016/0377-0427(87)90125-7. URL https://www.sciencedirect.com/science/article/pii/0377042787901257.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units, 2016. URL https://arxiv.org/abs/1508.07909.

Yuchen Shen and Barnabás Póczos. Graphbpe: Molecular graphs meet byte-pair encoding, 2024. URL https://arxiv.org/abs/2407.19039.

Nils Strodthoff, Patrick Wagner, Tobias Schaeffter, and Wojciech Samek. Deep learning for ecg analysis: Benchmarks and insights from ptb-xl. *IEEE Journal of Biomedical and Health Informatics*, 25:1519–1528, 2021.

Saedeh Tahery, Fatemeh Hamid Akhlaghi, Termeh Amirsoleimani, and Saeed Farzi. Heartbert: A self-supervised ecg embedding model for efficient and effective medical signal analysis, 2024. URL https://arxiv.org/abs/2411.11896.

Jialu Tang, Tong Xia, Yuan Lu, Cecilia Mascolo, and Aaqib Saeed. Electrocardiogram-language model for few-shot question answering with meta learning, 2024a. URL https://arxiv.org/abs/2410.14464.

Jialu Tang, Tong Xia, Yuan Lu, Cecilia Mascolo, and Aaqib Saeed. Electrocardiogram report generation and question answering via retrieval-augmented self-supervised modeling, 2024b. URL https://arxiv.org/abs/2409.08788.

Nazgol Tavabi and Kristina Lerman. Pattern discovery in time series with byte pair encoding, 2021. URL https://arxiv.org/abs/2106.00614.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, Anton Tsitsulin, Nino Vieillard, Piotr Stanczyk, Sertan Girgin, Nikola Momchev, Matt Hoffman, Shantanu Thakoor, Jean-Bastien Grill, Behnam Neyshabur, Olivier Bachem, Alanna Walton, Aliaksei Severyn, Alicia Parrish, Aliya Ahmad, Allen Hutchison, Alvin Abdagic, Amanda Carl, Amy Shen, Andy Brock, Andy Coenen, Anthony Laforge, Antonia Paterson, Ben Bastian, Bilal Piot, Bo Wu, Brandon Royal, Charlie Chen, Chintu Kumar, Chris Perry, Chris Welty, Christopher A. Choquette-Choo, Danila Sinopalnikov, David Weinberger, Dimple Vijaykumar, Dominika Rogozińska, Dustin Herbison, Elisa Bandy, Emma Wang, Eric Noland, Erica Moreira, Evan Senter, Evgenii Eltyshev, Francesco Visin, Gabriel Rasskin, Gary Wei, Glenn Cameron, Gus Martins, Hadi Hashemi, Hanna Klimczak-Plucińska, Harleen Batra, Harsh Dhand, Ivan Nardini, Jacinda Mein, Jack Zhou, James Svensson, Jeff Stanway, Jetha Chan, Jin Peng Zhou, Joana Carrasqueira, Joana Iljazi, Jocelyn Becker, Joe Fernandez, Joost van Amersfoort, Josh Gordon, Josh Lipschultz, Josh Newlan, Ju yeong Ji, Kareem Mohamed, Kartikeya Badola, Kat Black, Katie Millican, Keelin McDonell, Kelvin Nguyen, Kiranbir Sodhia, Kish Greene, Lars Lowe Sjoesund, Lauren Usui, Laurent Sifre, Lena Heuermann, Leticia Lago, Lilly McNealus, Livio Baldini Soares, Logan Kilpatrick, Lucas Dixon, Luciano Martins, Machel Reid, Manvinder Singh, Mark Iverson, Martin Görner, Mat Velloso, Mateo Wirth, Matt Davidow, Matt Miller, Matthew Rahtz, Matthew Watson, Meg Risdal, Mehran Kazemi, Michael Moynihan, Ming Zhang, Minsuk Kahng, Minwoo Park, Mofi Rahman, Mohit Khatwani, Natalie Dao, Nenshad Bardoliwalla, Nesh Devanathan, Neta Dumai, Nilay Chauhan, Oscar Wahltinez, Pankil Botarda, Parker Barnes, Paul Barham, Paul Michel, Pengchong Jin, Petko Georgiev, Phil Culliton, Pradeep Kuppala, Ramona Comanescu, Ramona Merhej, Reena Jana, Reza Ardeshir Rokni, Rishabh Agarwal, Ryan Mullins, Samaneh Saadat, Sara Mc Carthy, Sarah Cogan, Sarah Perrin, Sébastien M. R. Arnold, Sebastian Krause, Shengyang Dai, Shruti Garg, Shruti Sheth, Sue Ronstrom, Susan Chan, Timothy Jordan, Ting Yu, Tom Eccles, Tom Hennigan, Tomas Kocisky, Tulsee Doshi, Vihan Jain, Vikas Yadav, Vilobh Meshram, Vishal Dharmadhikari, Warren Barkley, Wei Wei, Wenming Ye, Woohyun Han, Woosuk Kwon, Xiang Xu, Zhe Shen, Zhitao Gong, Zichuan Wei, Victor Cotruta, Phoebe Kirk, Anand Rao, Minh Giang, Ludovic Peran, Tris Warkentin, Eli Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, D. Sculley, Jeanine Banks, Anca Dragan, Slav Petrov, Oriol Vinyals, Jeff Dean, Demis Hassabis, Koray Kavukcuoglu, Clement Farabet, Elena Buchatskaya, Sebastian Borgeaud, Noah Fiedel, Armand Joulin, Kathleen Kenealy, Robert Dadashi, and Alek Andreev. Gemma 2: Improving open language models at a practical size, 2024. URL https://arxiv.org/abs/2408.00118.

Akhil Vaid, Joy Jiang, Ashwin Sawant, Stamatios Lerakis, Edgar Argulian, Yuri Ahuja, Joshua Lampert, Alexander Charney, Hayit Greenspan, Benjamin Glicksberg, Jagat Narula, and Girish Nadkarni. Heartbeit: Vision transformer for electrocardiogram data improves diagnostic performance at low sample sizes, 2022.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

Patrick Wagner, Nils Strodthoff, Ralf-Dieter Bousseljot, Dieter Kreiseler, Fatima I. Lunze, Wojciech Samek, and Tobias Schaeffter. PTB-XL, a large publicly available electrocardiography dataset. *Scientific Data*, 7(1):154, May 2020. ISSN 2052-4463. doi: 10.1038/s41597-020-0495-6. URL https://www.nature.com/articles/s41597-020-0495-6. Number: 1 Publisher: Nature Publishing Group.

Zhongwei Wan, Che Liu, Xin Wang, Chaofan Tao, Hui Shen, Zhenwu Peng, Jie Fu, Rossella Arcucci, Huaxiu Yao, and Mi Zhang. Meit: Multi-modal electrocardiogram instruction tuning on large language models for report generation, 2024. URL https://arxiv.org/abs/2403.04945.

Sarah Wiegreffe and Yuval Pinter. Attention is not not explanation, 2019.

Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, 2 (1):37–52, 1987. ISSN 0169-7439. doi: https://doi.org/10.1016/0169-7439(87)80084-9. URL https://www.sciencedirect.com/science/article/pii/0169743987800849. Proceedings of the Multivariate Statistical Workshop for Geologists and Geochemists.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Huggingface's transformers: State-of-the-art natural language processing, 2020.

Sangmin Woo, Donguk Kim, Jaehyuk Jang, Yubin Choi, and Changick Kim. Don't miss the forest for the trees: Attentional vision calibration for large vision language models, 2024. URL https://arxiv.org/abs/2405.17820.

Shunxiang Yang, Cheng Lian, and Zhigang Zeng. Masked autoencoder for ecg representation learning. In *2022 12th International Conference on Information Science and Technology (ICIST)*, pages 95–98, 2022. doi: 10.1109/ICIST55546.2022.9926900.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. Opt: Open pre-trained transformer language models, 2022. URL https://arxiv.org/abs/2205.01068.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. *ArXiv*, abs/1904.09675, 2020.

Yubao Zhao, Tian Zhang, Xu Wang, Puyu Han, Tong Chen, Linlin Huang, Youzhu Jin, and Jiaju Kang. Ecg-chat: A large ecg-language model for cardiac disease diagnosis, 2024. URL https://arxiv.org/abs/2408.08849.

Appendix A. ECG-Byte

A.1. Additional Pseudocode for ECG-Byte

Algorithm 3: Merging a pair in an ID array

```
Input: Array of IDs ids, Pair to merge pair as (u_1, u_2), New ID new_id
Output: Merged vector of IDs
Function merge(ids, pair, new_id):
    Initialize empty vector new_ids with capacity of ids;
    i ← 0;
    while i < length of ids do
        if i < length of ids −1 and (ids[i], ids[i+1]) = pair then
            new_ids.append(new_id);
            i ← i + 2;
        end
        else
            new_ids.append(ids[i]);
            i ← i + 1;
        end
    end
    return new_ids;
```

Algorithm 4: Calculating Frequency of Byte Pairs in an Array

```
Input: Array of IDs ids
Output: HashMap of pairs and their frequencies
Function get_stats(ids):
    pair_counts ← Parallel fold operation:;
        foreach window of size 2 in ids do
            Let (u_1, u_2) ← elements of the window;
            Increment the count of (u_1, u_2) in local pair_count;
        end
    pair_counts ← Parallel reduce operation to combine local pair_count HashMaps;
    return pair_counts;
```

A.2. Mapping between Token and ECG

We add more examples of the mapping between the ECG signal and the encoded tokens for **ECG-Byte** in Figures 5 and 6.

A.3. Token usage and length distribution for varying `num_merges`

We add more examples of the token usage and length distributions for varying `num_merges` in Figure 7.

A.4. Attention Visualizations

We add more visualizations of the attention weights in Figures 8, 9, 10, 11, 12, 13, 14, 15.

Appendix B. 2-stage Pretraining Approaches

To be consistent, we normalize each ECG in the same manner as described in subsection 3.2. Consider a dataset of N ECG-image and clinical note pairs, denoted as $\{(I_i, O_i)\}_{i=1}^{N}$, where: $I_i \in \mathbb{R}^{3\times C\times T}$ is the i-th normalized and replicated ECG image, obtained by stacking the clipped ECG signal X_{clipped} along the channel dimension: $I_i = \text{stack}(X_{\text{clipped}}, X_{\text{clipped}}, X_{\text{clipped}})$. The reason we do this is because we need to create RGB images to use pretrained image models like ViT (Dosovitskiy et al., 2021) and CLIP (Radford et al., 2021).

O_i is the corresponding clinical note for the i-th ECG, serving as the textual description. Note that O_i differs from S in the autoregressive setup, where S represents the tokenized answer sequence provided by either ECG-QA (Oh et al., 2023) or MIMIC-IV ECG pretraining (Zhao et al., 2024).

Given these two features I and O we then describe the contrastive, masked, and dual approaches implemented for our baselines that are derived from commonly used techniques used throughout previous works (Oh et al., 2022; Choi et al., 2023; McKeen et al., 2024; Pham et al., 2024; Tang et al., 2024a,b; Vaid et al., 2022).

B.1. Contrastive learning approaches

We utilize a pretrained CLIP Radford et al. (2021) checkpoint, namely 'openai/clip-vit-base-patch32', provided by HuggingFace (Wolf et al., 2020) to encode ECG signals I and text labels O into a shared embedding space. Let $f_{\text{img}} : \mathbb{R}^{3\times C\times T} \to \mathbb{R}^d$ and

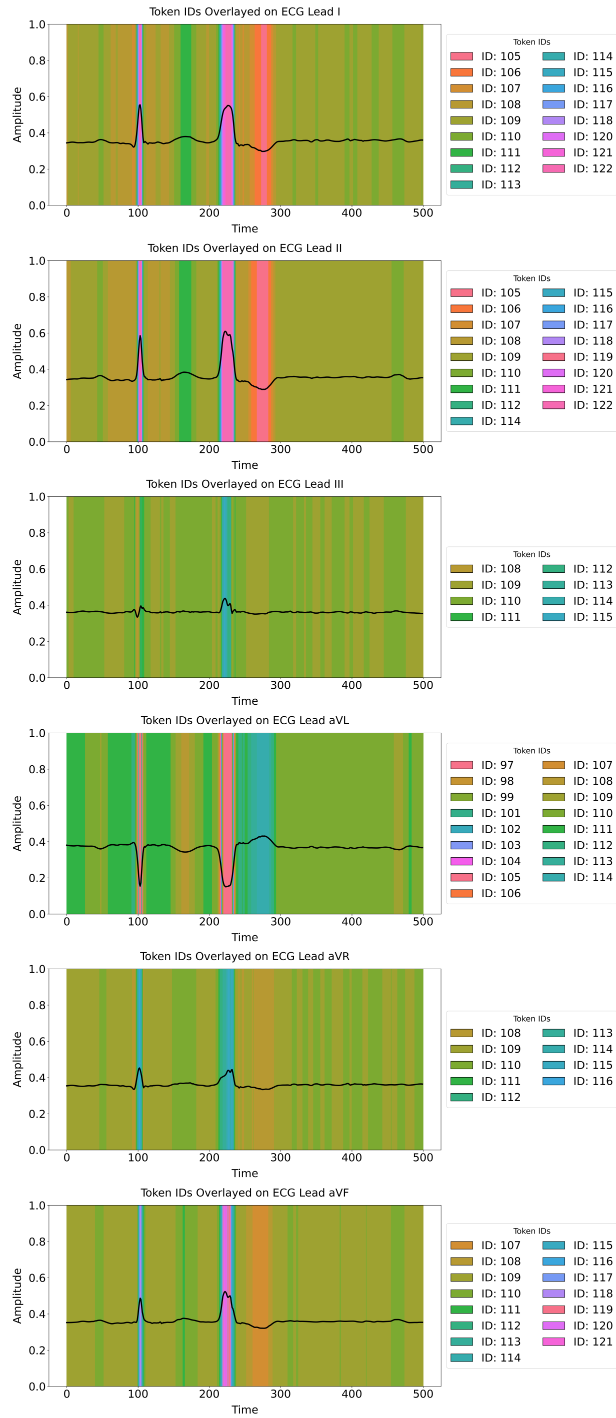

Figure 5: A mapping between tokens used for a given ECG Leads I, II, III, aVL, aVR, aVF.

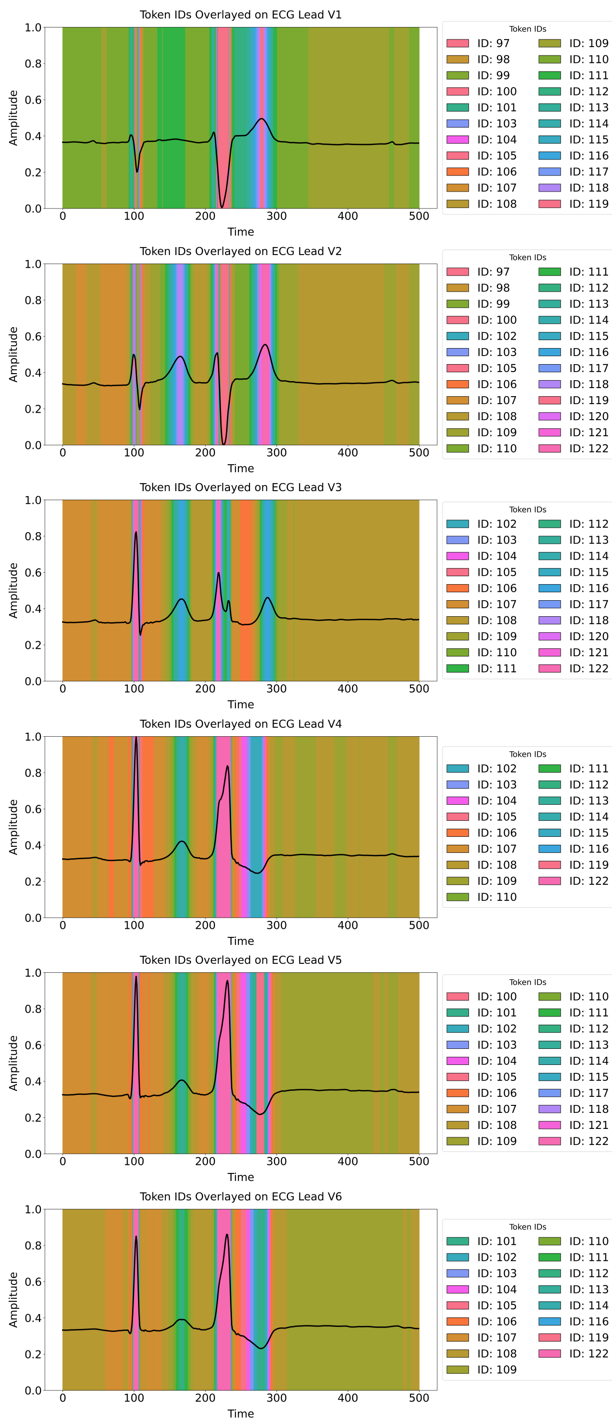

Figure 6: A mapping between tokens used for a given ECG Leads V1, V2, V3, V4, V5, V6.

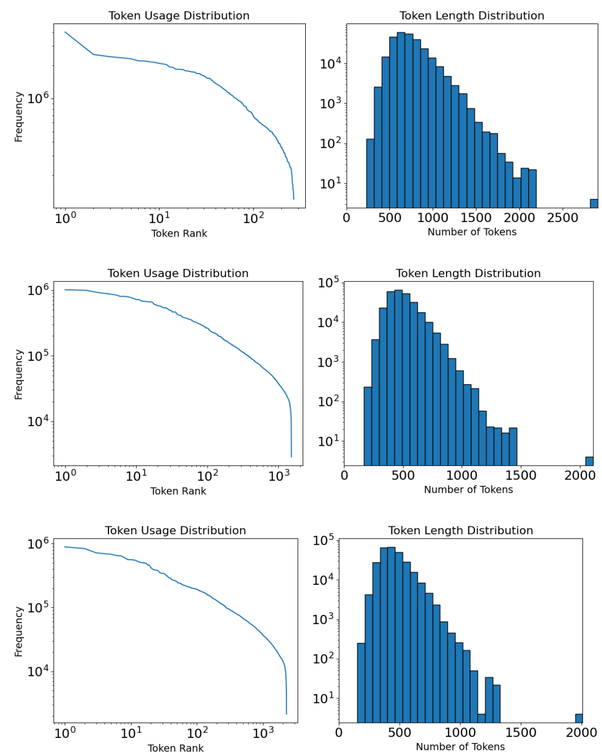

Figure 7: Plots of the token usage and length distributions for **ECG-Byte** where `num_merges` is 500, 1750, and 2500 from top to bottom.

Figure 8: The attention weight overlayed on both text (top) and ECG (bottom).

Figure 10: The attention weight overlayed on both text (top) and ECG (bottom).

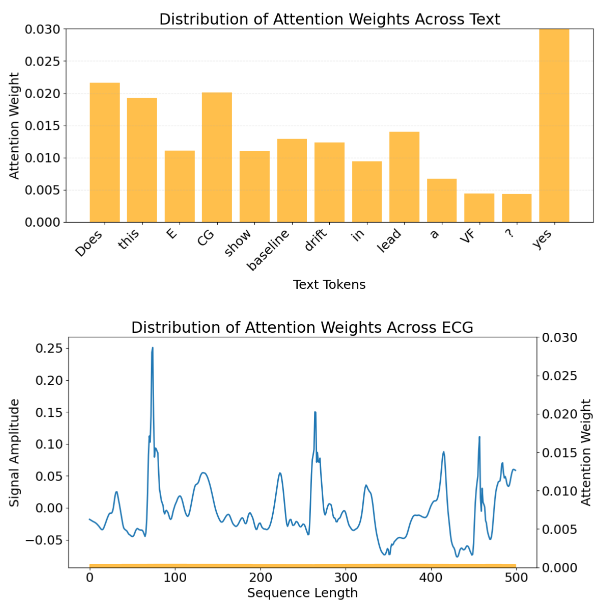

Figure 9: The attention weight overlayed on both text (top) and ECG (bottom).

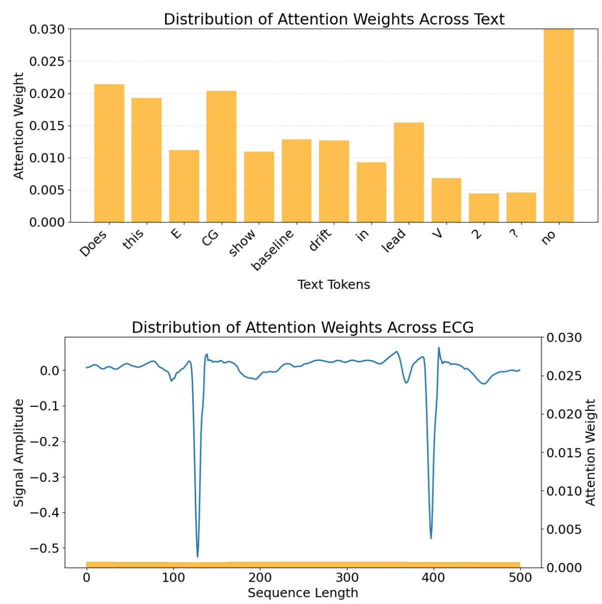

Figure 11: The attention weight overlayed on both text (top) and ECG (bottom).

Figure 12: The attention weight overlayed on both text (top) and ECG (bottom).

Figure 13: The attention weight overlayed on both text (top) and ECG (bottom).

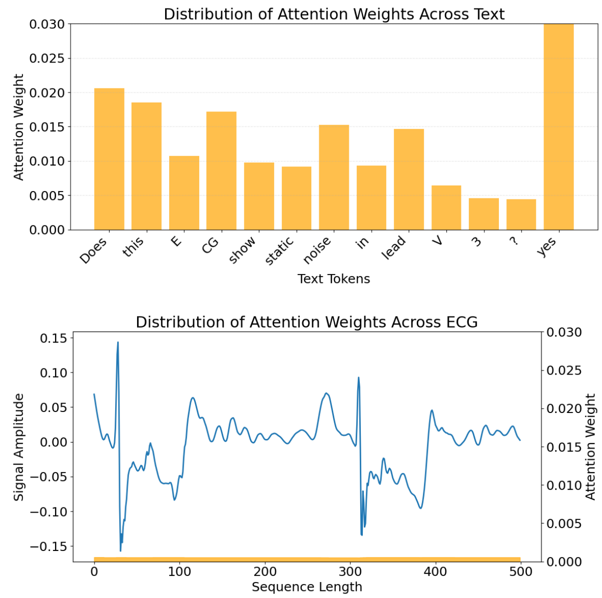

Figure 14: The attention weight overlayed on both text (top) and ECG (bottom).

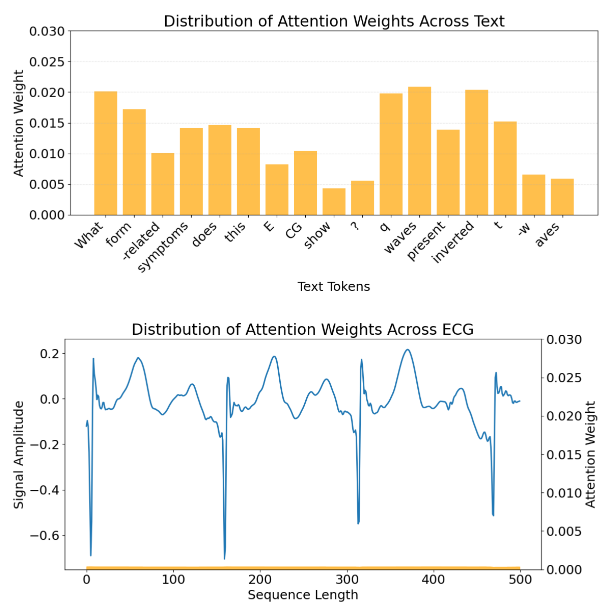

Figure 15: The attention weight overlayed on both text (top) and ECG (bottom).

f_{txt} : Text $\rightarrow \mathbb{R}^d$ be the image and text encoders of the pretrained CLIP model, respectively. The embeddings for the i-th pair are computed as:

$$z_i^{\text{img}} = f_{\text{img}}(I_i), \quad z_i^{\text{txt}} = f_{\text{txt}}(O_i),$$

where $z_i^{\text{img}}, z_i^{\text{txt}} \in \mathbb{R}^d$. The CLIP loss function $\mathcal{L}_{\text{CLIP}}$ aligns the embeddings of corresponding ECG signals and text labels while contrasting them with non-matching pairs. This is formulated as:

$$\mathcal{L}_{\text{CL}} = -\frac{1}{N}\sum_{i=1}^{N}\left[\log \frac{\exp\left(\text{sim}(z_i^{\text{img}}, z_i^{\text{txt}})/\tau\right)}{\sum_{j=1}^{N}\exp\left(\text{sim}(z_i^{\text{img}}, z_j^{\text{txt}})/\tau\right)} + \log \frac{\exp\left(\text{sim}(z_i^{\text{txt}}, z_i^{\text{img}})/\tau\right)}{\sum_{j=1}^{N}\exp\left(\text{sim}(z_i^{\text{txt}}, z_j^{\text{img}})/\tau\right)}\right]$$

where $\text{sim}(\cdot,\cdot)$ denotes cosine similarity, and τ is a learnable temperature parameter.

To integrate the pretrained CLIP model into our language model for joint reasoning over ECG signals and text, we project the frozen image embeddings z_i^{img} into the language model's hidden space. Let $W \in \mathbb{R}^{h\times d}$ be a learnable projection matrix, where h is the hidden dimension of the language model. The projected embeddings are:

$$z_i^{\text{clip}} = W z_i^{\text{img}}.$$

These projected embeddings z_i^{clip} are then prepended to the token embeddings of the language model, where we get $\texttt{Context} = \{\texttt{[BOS]}, \texttt{[SIG\_START]}, z_i^{\text{clip}}, \texttt{[SIG\_END]}, Q\}$ to train the same autoregressive objective, L_{NLL}.

B.2. Masked image modeling approaches

Consider the normalized ECG image $I \in \mathbb{R}^{3\times C\times T}$ obtained as previously described. We utilize a pretrained Vision Transformer (ViT) model (Dosovitskiy et al., 2021), specifically the 'google/vit-base-patch16-224-in21k' checkpoint provided by HuggingFace (Wolf et al., 2020).

The image I is partitioned into P non-overlapping patches. Let N be the number of images in our dataset, and I_i denote the i-th image. The ViT encoder f_{vit} projects these patches into latent embeddings:

$$z_i^{\text{patch}} = f_{\text{vit}}(I_i) \in \mathbb{R}^{P\times d},$$

where d is the embedding dimension of the ViT model.

During training, we randomly mask a subset of patches for each image I_i, creating a binary mask $M_i \in \{0,1\}^P$, where $M_{i,j} = 1$ if patch j is masked and $M_{i,j} = 0$ otherwise. The masked embeddings z_i^{masked} are formed by replacing the embeddings of masked patches with a mask token. A reconstruction head f_{rec} is then applied to predict the pixel-level content of the masked patches:

$$\hat{I}_i = f_{\text{rec}}(z_i^{\text{masked}}) \in \mathbb{R}^{P\times d}.$$

The masked image modeling loss $\mathcal{L}_{\text{MIM}}$ is computed as the mean squared error (MSE) between the reconstructed embeddings $\hat{I}_i$ and the original embeddings z_i^{patch} at the masked positions:

$$\mathcal{L}_{\text{MIM}} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{\sum_{j=1}^{P} M_{i,j}}\sum_{j=1}^{P} M_{i,j}\left\|\hat{I}_i[j] - z_i^{\text{patch}}[j]\right\|_2^2. \tag{7}$$

To integrate the MIM representations into the language model for joint reasoning over ECG signals and textual questions, we project the frozen ViT embeddings $z_i^{\text{img}} \in \mathbb{R}^d$ into the language model's hidden space. Let $W \in \mathbb{R}^{h\times d}$ be a learnable projection matrix, where h is the hidden dimension of the language model. The projected embeddings are given by:

$$z_i^{\text{vit}} = W z_i^{\text{img}}.$$

These projected embeddings z_i^{vit} are then prepended to the language model's token embeddings, to get $\texttt{Context} = \{\texttt{[BOS]}, \texttt{[SIG\_START]}, z_i^{\text{vit}}, \texttt{[SIG\_END]}, Q\}$ to train the same autoregressive objective, L_{NLL}, mentioned previously.

B.3. Dual approaches

The dual approach follows the previous two contrastive and masked image modeling approaches for pretraining the ECG encoder but simply just combines the losses like so:

$$\mathcal{L}_{\text{Dual}} = \lambda_1\mathcal{L}_{\text{MIM}} + \lambda_2\mathcal{L}_{\text{CL}}$$

where $\lambda_1 = \lambda_2 = 1$ in our study.

However, when training the autoregressive LLM, we project both embeddings, z_i^{vit} and z_i^{clip}, outputted by their respective frozen encoders via a learnable projection matrix into the language model's hidden space of dimension h. We then concatenate the projected embeddings and pass them through a fusion network to obtain the fused visual embedding

$z_i^{\text{fused}} \in \mathbb{R}^h$:

$$z_i^{\text{fused}} = f_{\text{fusion}}(\text{concat}(z_i^{\text{vit}}; z_i^{\text{clip}})),$$

where f_{fusion} is a trainable feedforward network. The fused visual embedding z_i^{fused} is prepended to the token embeddings of the language model, forming $\texttt{Context} = \{\texttt{[BOS]}, \texttt{[SIG\_START]}, z_i^{\text{fused}}, \texttt{[SIG\_END]}, Q\}$ to train the autoregressive objective, L_{NLL}.

Appendix C. Additional Results

C.1. Does Larger LLMs Yield Higher Performance?

We present the results of ablating the size of the LLM in Table 8. Interestingly, the performance across the three different model sizes (1B, 3B, 8B) remains fairly similar. We believe that the limited dataset size prevents the larger models from realizing their full performance potential. We hypothesize that increasing the amount of training data would enable the larger models to leverage their greater capacity, resulting in observable performance improvements.

Table 8: Ablation study on how larger LLMs perform for NLG.

LLM	BLEU-4	Rouge-L	Meteor	BertScore F1
Llama 3.2 1B (Grattafiori et al., 2024)	13.93 ± 0.21	**47.08** ± 0.56	29.17 ± 0.31	**92.53** ± 0.07
Llama 3.2 3B (Grattafiori et al., 2024)	14.80 ± 0.17	46.55 ± 0.21	29.53 ± 0.16	92.42 ± 0.01
Llama 3.1 8B (Grattafiori et al., 2024)	13.80 ± 0.16	46.29 ± 0.25	28.56 ± 0.11	92.44 ± 0.05

C.2. Qualitative NLG Examples

We provide qualitative NLG examples of successful (Figure 17) and unsuccessful generations (Figure 16).

Ground Truth Question	Which diagnostic symptom does this ECG show, incomplete left bundle branch block or incomplete right bundle branch block, excluding uncertain symptoms?	Does the qrs duration shown on this ECG fall within the normal range?	What form-related traits are exhibited by this ECG in lead I?	In lead V2, what form-related features does this ECG display?	What direction is this ECG deviated to?
Ground Truth Answer	incomplete right bundle branch block	yes	low amplitude t-wave	q waves present inverted t-waves	extreme axis deviation
Generated Answer	incomplete left bundle branch block	no	non-specific st depression	none	left axis deviation

Ground Truth Question	Within which numeric range does the qt interval of this ECG fall, above the normal range or within the normal range	Which diagnostic symptom does this ECG show, subendocardial injury in anterolateral leads or subendocardial injury in inferolateral leads, excluding uncertain symptoms?	Which diagnostic symptom does this ECG show, myocardial infarction in inferoposterolateral leads or myocardial infarction in anterolateral leads, excluding uncertain symptoms?	What form-related symptoms does this ECG show in lead II?	What diagnostic symptoms does this ECG show, excluding uncertain symptoms?
Ground Truth Answer	none	none	myocardial infarction in anterolateral leads	high qrs voltage	myocardial infarction in anteroseptal leads non-diagnostic t abnormalities
Generated Answer	qt interval	subendocardial injury in anterolateral leads	myocardial infarction in inferoposterolateral leads	non-specific st depression	myocardial infarction in anteroseptal leads

Figure 16: Randomly sampled NLG results of unsuccessful generations on the PTB-XL test set from ECG-QA.

Ground Truth Question	Is atrial fibrillation detectable from this ECG?	Which diagnostic symptom does this ECG show, left posterior fascicular block or subendocardial injury in lateral leads, including uncertain symptoms?	Which diagnostic symptom does this ECG show, subendocardial injury in lateral leads or incomplete left bundle branch block, including uncertain symptoms?	What is the diagnostic symptom that can be identified from this ECG, excluding any symptoms that are unclear, left atrial overload/enlargement or myocardial infarction in anterolateral leads?	What are the leads on the ECG that are manifesting static noise?
Ground Truth Answer	yes	none	subendocardial injury in lateral leads	left atrial overload/enlargement	lead I lead II lead III lead aVR lead aVL lead aVF lead V1 lead V2 lead V3 lead V4 lead V5 lead V6
Generated Answer	yes	none	subendocardial injury in lateral leads	left atrial overload/enlargement	lead I lead II lead III lead aVR lead aVL lead aVF lead V1 lead V2 lead V3 lead V4 lead V5 lead V6

Ground Truth Question	Does this ECG reveal any signs of sinus bradycardia?	Are there any noises detected in lead aVF on this ECG?	What numeric features of this ECG fall below the normal range?	What types of noises are displayed in lead aVL in this ECG waveform?	By excluding uncertain symptoms, which diagnostic symptom is apparent in this ECG, ischemic in inferior leads or left anterior fascicular block?
Ground Truth Answer	no	no	pr interval qt corrected qt interval	baseline drift static noise	left anterior fascicular block
Generated Answer	no	no	pr interval qt corrected qt interval	baseline drift static noise	left anterior fascicular block

Figure 17: Randomly sampled NLG results of successful generations on the PTB-XL test set from ECG-QA.