Typing Reinvented: Towards Hands-Free Input via sEMG

Kunwoo Lee
Mechanical Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
kunwool@andrew.cmu.edu

Dhivya Sreedhar
Information Systems Management
Carnegie Mellon University
Pittsburgh, PA 15213
dsreedha@andrew.cmu.edu

Pushkar Saraf
Information Systems Management
Carnegie Mellon University
Pittsburgh, PA 15213
psaraf@cmu.edu

Chaeeun Lee
Biomedical Engineering
Carnegie Mellon University
Pittsburgh, PA, 15213
chaeeunl@andrew.cmu.edu

Abstract

We explore surface electromyography (sEMG) as an alternative input modality for mapping muscle activity to keyboard inputs, aiming to enhance immersive typing experiences in next-generation human-computer interaction (HCI) systems. This approach is particularly relevant for emerging technologies such as spatial computing and virtual reality (VR), where traditional input methods are often impractical. By implementing transformer-based architectures, we achieved a substantial reduction in character error rate (CER) compared to a baseline model using time depth separable convolution (**online inference CER: 24.98% → 20.34%**). Our findings demonstrate the promise of transformer-based sEMG decoding and provide new insights into the development of real-time, muscle-driven interaction technologies.

1 Introduction

Since the invention of the typewriter in 1868, the basic design of typing devices—machines with buttons representing characters—has remained unchanged. While attempts to revolutionize keyboards exist, none have been transformative. As keyboards are crucial for translating human intent into computer input, it's worth reevaluating their relevance, especially as new computing formats, like Apple Vision Pro's spatial computing, emerge.

In this new era of computing, traditional keyboards may become obsolete, yet the instinct to press keys while typing remains deeply embedded in how we interact with technology. For instance, Vision Pro's gaze-based typing proves impractical for tasks like document creation. This highlights the need for virtual keyboards that preserve the familiar act of typing without relying on physical hardware or a desk—enabling input in mid-air through a floating interface. Our approach leverages sEMG to map muscle signals to virtual keystrokes, which is anticipated to be suitable in this context.

With this system, we aim to replicate the natural experience of typing by capturing fine-grained muscle activity, preserving users' cognitive and physical expectations. To validate our approach, we target state-of-the-art performance on the emg2qwerty dataset using modern neural architectures such as Transformers and Conformers. We evaluate model accuracy using character error rate (CER), benchmark against existing baselines, and explore latency and inference speed to assess the system's viability for real-time mid-air typing.

Preprint. Under review.

2 Related Work and Background

Early work on EMG-driven text input covered a range of topics and produced encouraging, but ultimately limited results, all constrained by the modest size of their data. An early brain-computer-interface study combined eye tracking with EMG signals to control a virtual keyboard; it showed that users could reliably select characters with mean accuracies above 90 %, yet the experiment involved only five participants and roughly 1 000 keystrokes in total, forcing the authors to rely on handcrafted thresholds rather than trainable models Dhillon et al. [2009]. A differential-EMG investigation of forearm muscles during sequential key presses explored how muscle activation varies with tempo and finger order, reporting clear, class-separable activation patterns and classification accuracies exceeding 85 %, but the study drew on fewer than 15 subjects and just minutes of data per task, precluding the use of data-hungry deep networks Chong et al. [2015]. Prototypes such as Air Keyboard and MyoKey tackled mid-air typing directly: Air Keyboard achieved about 7 words per minute with heuristic gesture mapping, while MyoKey reached 9 WPM using a shallow CNN; both systems, however, were trained on no more than a few thousand keystrokes from 10 users, making overfitting a central concern and limiting vocabulary size Gaba [2016]. Across these studies, the common thread is clear: despite promising proof-of-concept results, each investigation relied on ad-hoc datasets—typically under 20 minutes of sEMG per participant—rendering them inadequate for modern high-capacity sequence models such as Transformers or Conformers that thrive on large, diverse corpora.

As an effort to push the domain forward, Meta AI released two large-scale datasets -emg2qwerty and emg2pose.

- emg2qwerty: Contains non invasive emg signals collected from 108 users over 1335 sessions, over 346 hours making it the largest public dataset of its kind. Sivakumar et al. [2024b]
- emg2pose: Focuses on hand-based sEMG for estimating poses. The data set includes 193 users, 370 hours of data, and 29 stages featuring various gestures, offering a scale comparable to vision-based hand pose datasets. It contains 2kHz, 16-channel sEMG recordings paired with high-quality hand pose labels obtained from a 26-camera motion capture system. Salter et al. [2024]

For this project, we focus on the emg2qwerty dataset, with plans to incorporate the emg2pose dataset in the future to generate complementary virtual hand outputs for a more immersive environment. The size and richness of the emg2qwerty dataset have opened up new opportunities for researchers to develop innovative models and tools that advance the field. However, as a relatively new resource, it has yet to be fully explored, and no project have leveraged its full potential beyond baseline. Our work aims to address this gap by exploring advanced architectures, such as Transformers and Conformers, to surpass the current baseline and achieve state-of-the-art (SOTA) performance in sEMG-to-keyboard input mapping.

3 Dataset

The dataset in Sivakumar et al. [2024a] is currently the largest dataset for sEMG labeled by keyboard input as in Fig. 1. It comprises 1,135 sessions, 108 users, and 346 hours of recordings. From this dataset, Sivakumar et al. [2024a] demonstrated that with minimal investment, high performance can be achieved by leveraging existing architectures from related domains. Specifically, Time Depth Separable Convolution (TDS), a model popular in Automatic Speech Recognition (ASR) was used and achieved CER below 10%. This is motivating for us as there may be large room of improvement, potentially paving the way for the next generation of typing experiences.

The dataset contains 32 sEMG input channels—16 per hand—distributed around the wrists. It is intuitive that channels positioned close together will exhibit correlated signals, while those farther apart (e.g., two or more channels away) should be largely independent. However, due to variability in wristband placement between users, slight misalignments can introduce unexpected correlations between more distant channels. To quantify this effect, we plotted the inter-channel correlation matrix, as shown in Fig. 2. The results show that adjacent channels often display strong correlation, with influence extending up to two neighboring channels. This finding suggests the importance of incorporating rotational invariance into the model to ensure robustness across different users.

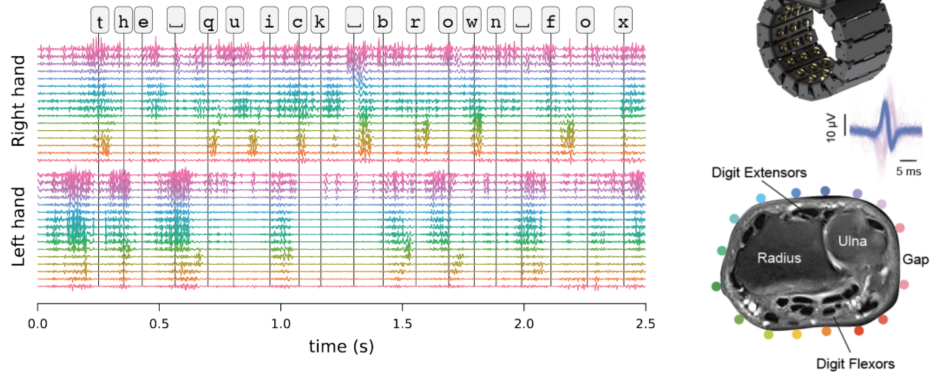


Figure 1: Dataset overview (Sivakumar et al. [2024a])

Moreover, the original emg2qwerty paper uses log-spectrogram representations as input to a Time-Delay Neural Network (TDS) encoder. To qualitatively assess the effectiveness of their preprocessing approach, we visualized several sample log-spectrograms (Fig. 3) and reviewed whether the chosen parameters adequately capture the relevant sEMG signal characteristics.

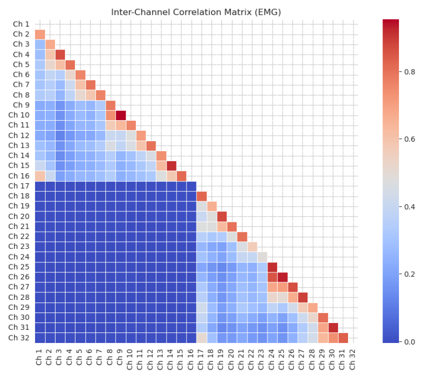


Figure 2: Inter-channel correlation

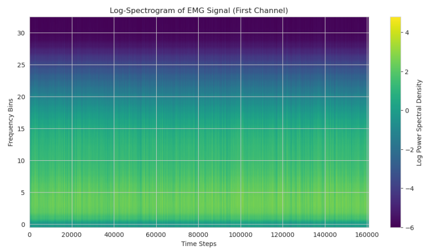


Figure 3: Log spectrogram of the signal

Based on our observations of the data, we applied several data augmentation techniques to enhance model generalization and training robustness.

- **TemporalAlignmentJitter:** Introduces small temporal shifts to simulate misalignments caused by hardware latency, improving the model's robustness to temporal variation.
- **RandomBandRotation:** Randomly rotates EMG channel order to simulate variations in wristband placement across users, encouraging rotational invariance.
- **SpecAugment:** Applies frequency masking, time masking, and time warping to the log-spectrograms, promoting generalization by preventing overfitting to specific temporal or spectral patterns.

4 Method

4.1 Baseline Model

We adopted the baseline model proposed by Sivakumar et al. [2024a] for the baseline model training, as there is no other work that has addressed this specific application with this dataset. It was inspired by the ASR modeling approach as their tasks resemble each other in that they take continuous waveform signals as input and map to a sequence of tokens. A 32-channel electrode array captures continuous muscle activity at a 2kHz sample rate, and the input is a 1D time series signal for each channel. The model employs analogous spectral features on a log-frequency scale tailored for sEMG. A 2D batch normalization is applied as the first layer to normalize across electrode channels.

To enhance generalization across a user's sessions and the band's placement, the model applies SpecAugment (Park et al. [2019]) with the additional forms of data augmentation, which are, electrode rotation augmentation – shifts all channels by -1, 0, or +1 position simulating electrode displacement, and temporal alignment jitter – random misalignment between left and right hand sEMG signals by 0-120 samples. (i.e., 0-60ms) With the same token, before the channel data is passed into the main model, each band of electrodes undergoes a small transformation called "Rotation-Invariance" module to compensate for the rotation shifts. Each module consists of a linear layer and a ReLU activation, and outputs for shits of -1, - and +1 are averaged over. The model architecture consists of a stock of convolutional blocks.

The main model adopts TDS developed by Hannun et al. [2019], which improves parameter efficiency while maintaining wide receptive fields. The receptive field is 1 second long to capture the muscle activity related to bigrams or trigrams, as co-articulation is dominant in sEMG activity. The TDS network outputs a sequence of encoded feature representations. emg2qwerty includes precise key-press and key-release events, which allows for the use of a cross-entropy classification loss averaged over all time points in the output sequence. However, CTC (Graves et al. [2006]) was found to work the best with this model empirically. A character-level 6-gram Kneser-Ney Language Model (LM) (Heafield et al. [2013]) was used to refine predictions. It is integrated with the CTC logits first-pas beam-search decoder to convert logits into character sequences. Unlike ASR decoding, emg2qwerty can delete the signal history using backspace, making it challenging, which led to implementing a modified backspace-aware beam-search decoder.

The proposed baseline focuses on acausal approach, incorporating data from both past and future frames, with a time window of 900ms into the past and 100ms into the future. While this can be effective for achieving high accuracy, this may limit the real-world usage. Thus for all of our models, we aim to make all approaches causal, only leveraging past data.

4.2 Proposed Models

The log spectrogram of the input data, following augmentation, is first processed by a spectrogram normalization layer, which reduces variability across users, environments, and recording devices. This normalization step is crucial for stabilizing training and improving model generalization. The normalized features are then passed through a multi-band rotationally invariant MLP, which acts as an adaptation layer to address potential misalignments in sEMG probe placement relative to an ideal configuration. This mechanism helps mitigate user-specific and inter-session variability.

After this sequence of preparatory layers, the processed sEMG data is fed into the encoder, which is responsible for extracting keystroke-relevant information. In the baseline model, this encoder is composed of TDS layers. In this project, we aim to enhance this baseline by proposing two alternative encoder architectures: the TDS-Transformer Encoder and the Conformer Encoder. The overall structure of our model is shown in Fig. 4.

The theme of two different approaches is leveraging transformer-based architecture and its variant. Transformers, as demonstrated in the work Vaswani et al. [2017], have proven to be exceptional in various tasks involving sequential data, such as natural language processing and multimodal learning. Our task involves mapping a sequential input of physiological signal that is continuous and temporal to keyboard input, which is discrete with linguistic or symbolic domain representation. Thus, transformers are expected to be suitable for our application. Further, variants such as Conformer Gulati et al. [2020], which integrate convolutional modules into the Transformer architecture, are particularly well-suited for modeling local temporal patterns in physiological signals while preserving the ability to capture long-range dependencies.

4.2.1 Approach 1) TDS - Transformer Encoder

We added a Transformer Encoder layer in the baseline model. [Fig. 4] Vaswani et al. [2017] proposed the Transformer architecture, which relies entirely on self-attention mechanisms to model relationships in sequential data without using recurrence. This architecture has since become a

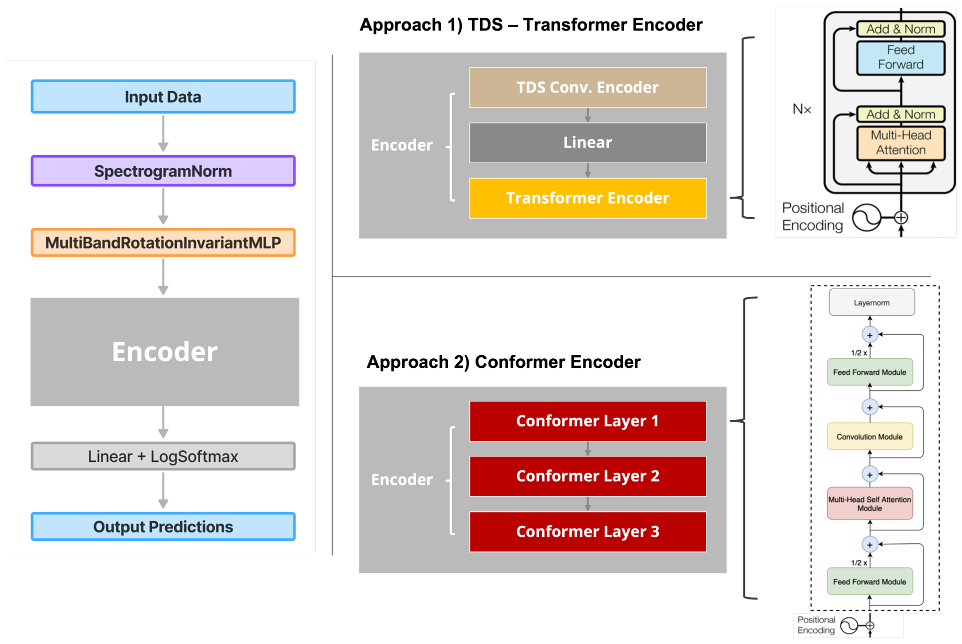


Figure 4: Proposed models diagram

standard in many ASR tasks due to its ability to capture long-range dependencies and contextual patterns effectively.

To integrate the Transformer into our pipeline, we placed it after the convolutional layers that process the EMG signals. A linear projection layer was introduced to reduce the output feature dimension from the TDS Encoder by a factor of four (from 768 to 192) before feeding it into the Transformer. Since our ablation budget and resources were limited, we conducted a focused set of experiments that yielded several actionable insights. We experimented with various reduction sizes and found that this 4× compression offered the best performance, likely serving as a bottleneck that encourages the model to retain only the most salient information from the EMG signal. Additionally, while increasing the number of Transformer layers or attention heads did not result in further gains, we consistently observed that keeping the feedforward network (FFN) dimension above 2000 was beneficial, suggesting that a more expressive FFN helps capture the nonlinear structure of EMG data.

The addition of the Transformer Encoder consistently reduced CER in the generic model. Its self-attention mechanism enabled the model to focus on the most informative temporal regions, effectively capturing long-range dependencies within the EMG signal. This helped disambiguate similar patterns by leveraging contextual cues.

4.2.2 Approach 2) Conformer Encoder

As illustrated in Fig. 4, we adopt a 3-layer Conformer encoder with 6 attention heads in our model. In the original Conformer work Gulati et al. [2020], the authors propose a 16-layer, 4-head architecture tailored for Automatic Speech Recognition (ASR) tasks, where input sequences are typically long and consist of single-channel audio. In contrast, our sEMG dataset consists of relatively short sequences (4 seconds) with a much higher input dimensionality: 32 channels in total, corresponding to 16 sensors on each hand. Given this structural difference, directly applying the original configuration is neither efficient nor optimal.

To address this, we significantly reduce the number of layers in the encoder, as deeper networks are generally beneficial for capturing long-range temporal dependencies, which are less critical in our shorter input sequences. At the same time, we increase the number of attention heads to better leverage the multi-channel nature of the data. More attention heads allow the model to attend to interactions between different sEMG channels, which is essential for capturing the spatial relationships and cross-channel dynamics introduced by variability in sensor placement and muscle activation patterns.

This adaptation ensures that the Conformer architecture is better aligned with the characteristics and demands of our physiological input modality.

Approach	Details
TDS Encoder (Baseline)	$d_{\text{in}} = 528$, 4 blocks, 24 channels, kernel size = 32, MLP dim = 384
TDS-Transformer Encoder	$d_{\text{in}} = 192$, 2 layers, 4 heads, kernel size = 32, FFN dim = 2048
Conformer Encoder	$d_{\text{in}} = 528$, 3 layers, 6 heads, kernel size = 31, FFN dim = 256

Table 1: Model architectures overview

4.3 Decoding Mechanism and LM

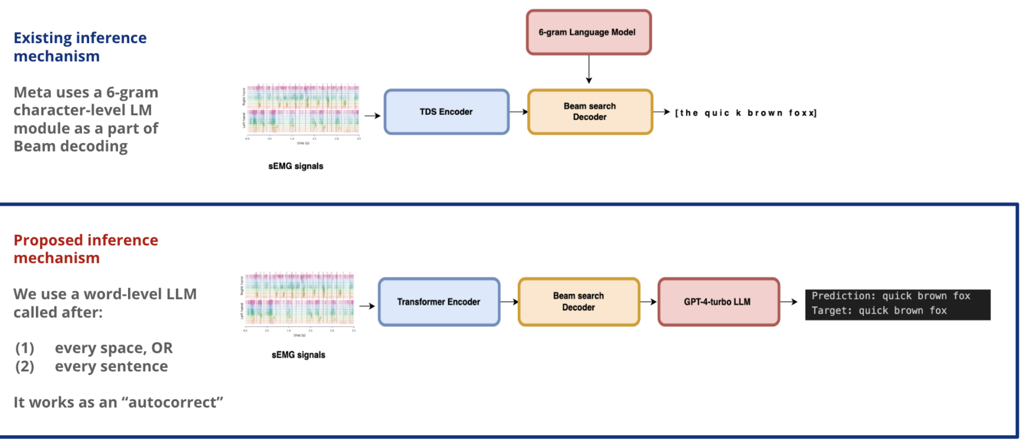

Existing inference mechanism
Meta uses a 6-gram character-level LM module as a part of Beam decoding
sEMG signals
TDS Encoder
6-gram Language Model
Beam search Decoder
[the quic k brown foxx]
Proposed inference mechanism
We use a word-level LLM called after:
(1) every space, OR
(2) every sentence
It works as an "autocorrect"
sEMG signals
Transformer Encoder
Beam search Decoder
GPT-4-turbo LLM
Prediction: quick brown fox
Target: quick brown fox

Figure 5: Proposed inference mechanism

As shown in Fig. 5, our decoding pipeline incorporates multiple strategies, enhanced with language model capabilities, to refine the encoder's raw predictions. This is particularly important for our virtual keyboard typing application, where users interact without physical keys. Even individuals confident in their touch-typing skills are susceptible to errors due to the lack of tactile feedback—regardless of the model's accuracy. The language model therefore acts as an intelligent auto-correction mechanism, improving overall input reliability.

We evaluated several decoding approaches of increasing complexity. Greedy Decoding serves as our baseline decoding method, representing the simplest approach where the most probable character is selected at each step without considering future consequences. Beam Search with n-gram LM, which incorporates a character-level 6-gram language model that provides statistical priors about character sequences, maintaining multiple hypothesis paths during decoding to find globally optimal solutions. We also experimented with integrating Flan-T5-small directly into the beam search itself. Using Flan-T5 inside the beam helped guide search decisions based on lightweight language modeling at each step. This approach leverages Flan-T5's strong few-shot performance capabilities even when compared to much larger models, allowing for efficient decision-making during the decoding process. For Word-Level LLM Integration, we experimented with two variations of integrating more sophisticated language models: correction after each space, where the beam search decoder invokes an LLM (such as Flan-T5-small or GPT-4-turbo) whenever a space is encountered, allowing word-by-word refinement; and sentence-level correction, where the decoder completes entire sentences before invoking the LLM for contextual refinement, reducing the frequency of API calls.

We experimented with Flan-T5-small, a compact transformer-based language model that offers an excellent balance between efficiency and performance, making it ideal for constrained deployment scenarios where computational resources are limited. Additionally, we experimented with more powerful models like GPT-2 and GPT-4-turbo to evaluate the trade-offs between computational requirements and correction quality, as these larger models demonstrate superior multi-task accuracy and reasoning capabilities but require significantly more computational resources.

4.4 Loss Function and Evaluation Metrics

We adopt the *Connectionist Temporal Classification* (CTC) loss for training, as it is well-suited for sequence prediction tasks where the input and output sequences are aligned in order but not one-to-one in time. This is particularly appropriate for our task of mapping continuous sEMG signals to discrete keystroke sequences. CTC is especially compatible with transformer-based architectures and offers several advantages:

- It accommodates the variable-length nature of both the input sEMG signals and output keystroke sequences.
- It enables the model to learn alignments automatically, without requiring frame-level labels or precise timing information for each character.
- It integrates smoothly with the transformer encoder output, allowing end-to-end training without additional alignment steps.

Formally, given an input sequence $\mathbf{x} = (x_1, x_2, \ldots, x_T)$ of length T, and a target sequence $\mathbf{y} = (y_1, y_2, \ldots, y_U)$ of length U, CTC defines the loss as the negative log-likelihood of the target sequence given all valid alignments $\mathcal{B}^{-1}(\mathbf{y})$:

$$\mathcal{L}_{\text{CTC}} = -\log P(\mathbf{y} \mid \mathbf{x}) = -\log \sum_{\pi \in \mathcal{B}^{-1}(\mathbf{y})} P(\pi \mid \mathbf{x}),$$

where π is a path over the extended alphabet (including a blank symbol), and $\mathcal{B}$ is the collapsing function that removes repeated symbols and blanks from π to produce the final output sequence $\mathbf{y}$.

For evaluation, we use the *Character Error Rate* (CER) as the primary metric. CER is defined as the normalized Levenshtein distance (edit distance) between the predicted sequence $\hat{\mathbf{y}}$ and the ground truth $\mathbf{y}$:

$$\text{CER} = \frac{S + D + I}{N},$$

where:

- S = number of substitutions,
- D = number of deletions,
- I = number of insertions,
- N = total number of characters in the ground truth sequence.

In addition to reporting the overall CER, we analyze its individual components—substitutions, insertions, and deletions—to gain deeper insight into the model's behavior and identify common error modes.

5 Results and Discussion

5.1 Baseline Results

We conducted training and testing on the baseline model to validate our proposed approach. The results are presented in Table 2. As shown in the first row, the personalized, acausal version of the model achieves a notably low character error rate (CER), whereas the generic version performs significantly worse. This performance gap reflects a well-known limitation of EMG-based systems: high inter-user variability often necessitates personalization for effective deployment.

In this work, we define online inference as the process of sliding a 4-second input window over the data and generating short, successive outputs that are later concatenated. In contrast, offline inference refers to feeding the entire signal into the model at once to generate a full paragraph. Contrary to the common assumption that online inference typically yields lower accuracy, we find that offline inference introduces a greater distribution shift and inference burden, resulting in a higher CER. This is likely due to the model having to handle longer temporal dependencies and more variable input-output mappings in a single forward pass.

The original baseline model is acausal and thus unsuitable for real-time applications. To enable practical deployment, we adapted it by replacing the acausal dataloader with a causal version and retrained the model. Surprisingly, the causal model exhibited only a marginal increase in CER compared to the acausal version. We attribute this robustness to two key factors. First, keystroke prediction at a given moment relies primarily on past and present EMG signals, with limited benefit from brief future context. While larger lookahead windows might offer gains, they would compromise real-time applicability. Second, due to the physiological property of sEMG signals preceding motion by tens of milliseconds, the model inherently captures future intent. Consequently, additional future data contributes little to performance. Furthermore, in our 4-second input window for online inference, only the final frames stand to benefit from future context, leaving the majority of the window unaffected.

Model	Generic CER (% Online / Offline)	Personalized CER (% Online / Offline)
TDS (baseline, acausal)	– / 55.38	– / 10.86
TDS (baseline, causal)	24.98 / 58.32	– / –

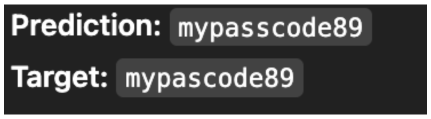
Table 2: Comparison of Character Error Rates (CER) for baseline models

5.2 Final Results

Results in table 3 highlight the performance of various models in terms of Character Error Rate (CER) under online and offline inference conditions. The baseline model, based on TDS-Conv architecture yields a cer of 24.98% for online inference and 58.32% for offline inference. Augmenting this model with more sophisticated architectures improves the performance. The transformer based model which retains the TDS front-end and adds a 4 layer transformer encoder, reduces the online/offline CER to 21.60%/44.43% respectively. Further improvements are seen when the existing architecture is replaced with a 3-layer conformer based architecture achieving the lowest cer of 20.34% (online) and 43.21% (offline). Personalized evaluation results are marked to be updated in the immediate future due to PSC outage. These results suggest that architectural choices, especially incorporating attention based encoders, significantly ehance model robustness. The Conformer outperforms the TDS-Transformer by seamlessly integrating both local and global feature modeling within a unified inference mechanism, whereas the TDS-Transformer relies on a more sequential approach, extracting local and global patterns in separate stages.

Model	Decoder	LM	Generic CER (Online / Offline)	Personalized CER (Offline)
TDS (baseline, causal)	Greedy	None	24.98 / 58.32	–
TDS + Transformer Enc.	Greedy	None	21.60 / 44.43	–
Conformer	Greedy	None	**20.34 / 43.21**	–
TDS (baseline, causal)	Beam	6-gram Char LM (baseline)	–	45.55
TDS (baseline, causal)	Beam	Flan-T5 (space-level)	–	28.55
TDS (baseline, causal)	Beam	Flan-T5 (sentence-level)	–	42.65
TDS (baseline, causal)	Beam	GPT-4 Turbo (space-level)	–	28.11
TDS (baseline, causal)	Beam	GPT-4 Turbo (sentence-level)	–	27.65
TDS (baseline, causal)	Greedy	Flan-T5 (space-level)	–	28.55
TDS (baseline, causal)	Greedy	Flan-T5 (sentence-level)	–	42.65
TDS (baseline, causal)	Greedy	GPT-4 Turbo (space-level)	–	24.98
TDS (baseline, causal)	Greedy	GPT-4 Turbo (sentence-level)	–	**22.71**

Table 3: Combined CER results by model and decoding strategy. Greedy decoding is performed without a language model (LM); Beam decoding integrates external LMs (6-gram Char LM, Flan-T5, or GPT-4 Turbo). CER = Character Error Rate (%). Personalized CER is reported for *offline* mode only.


Prediction: mypasscode89

Target: mypascode89

Figure 6: Sample prediction from gpt-4-turbo

Results in table 3 shows the performance of various language models with personalized decoding strategies. Among the personalized decoding strategies, the best performance is achieved using

GPT-4 Turbo with sentence-level correction during greedy decoding, yielding a CER of 22.71%, outperforming all other setups. While Flan-T5-small performs moderately well with word-level correction under greedy decoding (28.55%), it deteriorates under beam search, indicating its limitations in handling hypothesis diversity. Interestingly, beam search consistently under-performs greedy decoding when combined with LLM-based corrections, suggesting that the added complexity introduces mis-alignments that smaller or even larger LMs struggle to correct effectively.

While language model–based corrections are essential in our virtual keyboard setup, they come with trade-offs. Large models like GPT-4 Turbo enhance fluency and produce more natural text, making them well-suited for tasks such as document writing. However, in scenarios requiring exact reproduction—such as Fig. 6—they can introduce unintended modifications, like replacing "usr123!" with "user123." This illustrates the tension between improving readability and maintaining accuracy in sensitive inputs like passwords or usernames.

6 Future Directions

Although our approach achieves a lower Character Error Rate (CER) than the baseline model, several limitations remain that warrant further investigation. The most significant challenge is inference latency. Under our current online inference setup, users must wait 4 seconds before the first keystroke appears on the screen—a delay that is clearly unsuitable for real-world applications. Ideally, the end-to-end latency should be under 50 ms, which matches the response time of a typical Bluetooth keyboard.

While comparing a neural network-based system to a hardware device may seem unfair, achieving similar responsiveness is not entirely out of reach. One potential solution is to drastically reduce the sliding window size while maintaining a longer context window through padding. For instance, using a 50 ms sliding window with a 4-second historical padding window would allow the model to retain the same input size while enabling real-time updates. In our current setup, each inference takes roughly 50 ms, which would result in an effective latency of approximately 100 ms. Though still higher than hardware latency, this is a meaningful step forward.

Moreover, we can leverage domain knowledge in EMG, where muscle activation typically precedes physical movement by tens of milliseconds. This opens the door to predictive modeling, where the system anticipates keystrokes before they occur, potentially achieving less than 50 ms latency. That said, even our personalized models currently yield around 10% CER, which suggests the technology is not yet ready for deployment in real-world scenarios. Nonetheless, reducing latency and increasing accuracy are critical next steps toward practical, muscle-driven typing systems.

7 Conclusions

In this work, we proposed a transformer-based sEMG decoding framework to enable hands-free, muscle-driven typing for immersive computing environments. Building on the emg2qwerty dataset, we demonstrated that attention-based models, particularly Conformers, offer significant improvements in accuracy over the existing TDS-based baseline, reducing online CER from 24.98% to 20.34%. We also adapted the original acausal model to a causal format suitable for real-time inference, with minimal performance degradation, reinforcing the feasibility of practical deployment.

Despite these advancements, several challenges remain. Notably, the current system incurs an inference latency of several hundred milliseconds, which remains higher than conventional hardware input devices. However, by leveraging the anticipatory nature of sEMG signals and refining the model for predictive inference, under 100 ms latencies may be achievable in future iterations.

Our findings mark a step forward in developing intuitive, non-invasive input systems and suggest a promising path toward real-time, sEMG-driven interfaces in spatial and virtual computing. As computing continues to shift beyond screens and keyboards, we envision muscle-based interaction playing a central role in the next generation of human-computer interaction.

8 Code

GitHub Repository

References

Hyun Ju Chong, Soo Ji Kim, and Ga Eul Yoo. Differential effects of type of keyboard playing task and tempo on surface emg amplitudes of forearm muscles. *Frontiers in Psychology*, 6:1277, 2015.

Hari Singh Dhillon, Rajesh Singla, Navleen Singh Rekhi, and Rameshwar Jha. Eog and emg based virtual keyboard: A brain-computer interface. In *2009 2nd IEEE international conference on computer science and information technology*, pages 259–262. IEEE, 2009.

Jacob A Gaba. Air keyboard: Mid-air text input using wearable emg sensors and a predictive text model. 2016.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, page 369–376, New York, NY, USA, 2006. Association for Computing Machinery. ISBN 1595933832. doi: 10.1145/1143844.1143891. URL `https://doi.org/10.1145/1143844.1143891`.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, et al. Conformer: Convolution-augmented transformer for speech recognition. *arXiv preprint arXiv:2005.08100*, 2020.

Awni Hannun, Ann Lee, Qiantong Xu, and Ronan Collobert. Sequence-to-sequence speech recognition with time-depth separable convolutions. In *Interspeech 2019*, pages 3785–3789, 2019. doi: 10.21437/Interspeech.2019-2460.

Kenneth Heafield, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn. Scalable modified Kneser-Ney language model estimation. In Hinrich Schuetze, Pascale Fung, and Massimo Poesio, editors, *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 690–696, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL `https://aclanthology.org/P13-2121/`.

Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. Specaugment: A simple data augmentation method for automatic speech recognition. *arXiv preprint arXiv:1904.08779*, 2019.

Sasha Salter, Richard Warren, Collin Schlager, Adrian Spurr, Shangchen Han, Rohin Bhasin, Yujun Cai, Peter Walkington, Anuoluwapo Bolarinwa, Robert Wang, Nathan Danielson, Josh Merel, Eftychios Pnevmatikakis, and Jesse Marshall. emg2pose: A large and diverse benchmark for surface electromyographic hand pose estimation, 2024. URL `https://arxiv.org/abs/2412.02725`.

Viswanath Sivakumar, Jeffrey Seely, Alan Du, Sean R Bittner, Adam Berenzweig, Anuoluwapo Bolarinwa, Alexandre Gramfort, and Michael I Mandel. emg2qwerty: A large dataset with baselines for touch typing using surface electromyography. *arXiv preprint arXiv:2410.20081*, 2024a.

Viswanath Sivakumar, Jeffrey Seely, Alan Du, Sean R Bittner, Adam Berenzweig, Anuoluwapo Bolarinwa, Alexandre Gramfort, and Michael I Mandel. emg2qwerty: A large dataset with baselines for touch typing using surface electromyography, 2024b. URL `https://arxiv.org/abs/2410.20081`.

A Vaswani, N Shazeer, N Parmar, J Uszkoreit, L Jones, A Gomez, L Kaiser, and I Polosukhin. Attention is all you need. In *NIPS*, 2017. URL `https://doi.org/10.48550/arXiv.1706.03762`.