CapFM: A Foundation Model for Capacitive Touch Sensing

Rachel Schuchert
Department of Computer Science
ETH Zürich
Zurich, Switzerland
rachel.schuchert@inf.ethz.ch

Christian Holz
Department of Computer Science
ETH Zürich
Zurich, Switzerland
christian.holz@inf.ethz.ch

Abstract

Capacitive touch sensing supports rich interactive behavior, yet learning-based methods have been task-specific and rely on labeled datasets that are costly to collect and annotate. We introduce *CapFM*, a foundation model for capacitive touch sensing that learns a shared representation across diverse touch understanding tasks. We pretrain a Masked Autoencoder on a large corpus of unlabeled capacitive frames across seven datasets, through which our model captures a robust structure in the signal without task-specific supervision. For downstream use, we keep CapFM's pretrained backbone fixed and attach lightweight task heads that use either global embeddings or spatial token features, depending on task output structure. We evaluate CapFM's unified approach on eight diverse tasks, including finger identification, pitch and yaw estimation of the finger angle, 3D hand-pose estimation, high-resolution contact area segmentation, and input accuracy. Across these settings, CapFM transfers effectively across datasets and hardware configurations. Using the same training and validation regimen, CapFM pretrained without the respective dataset matches or beats all baselines and shows robust performance. Our results show that foundation-model pretraining makes capacitive sensing models more data-efficient for downstream learning and transferable across tasks and hardware, which can support diverse applications based on CapFM's pretrained backbone rather than separate task-specific models.

CCS Concepts

• **Computing methodologies** → *Machine learning*; • **Human-centered computing** → **Touch screens**.

Keywords

Capacitive Sensing, Foundation Models, Self-Supervised Learning, Expressive Touch, Masked Autoencoders, Multi-Task Learning

ACM Reference Format:
Rachel Schuchert and Christian Holz. 2026. CapFM: A Foundation Model for Capacitive Touch Sensing. In *The 39th Annual ACM Symposium on User Interface Software and Technology (UIST '26), November 02–05, 2026, Detroit, MI, USA.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3830398.3830703

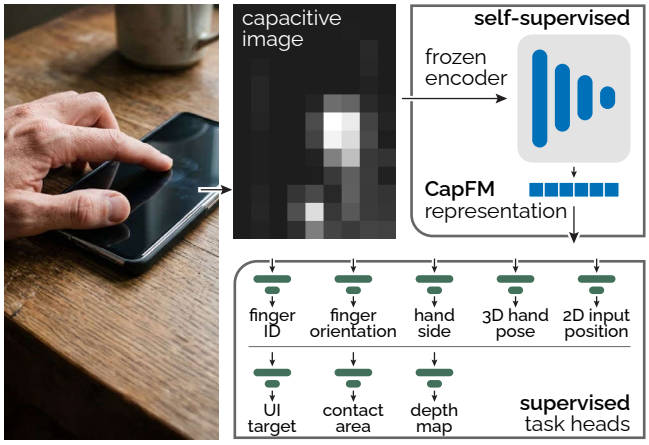


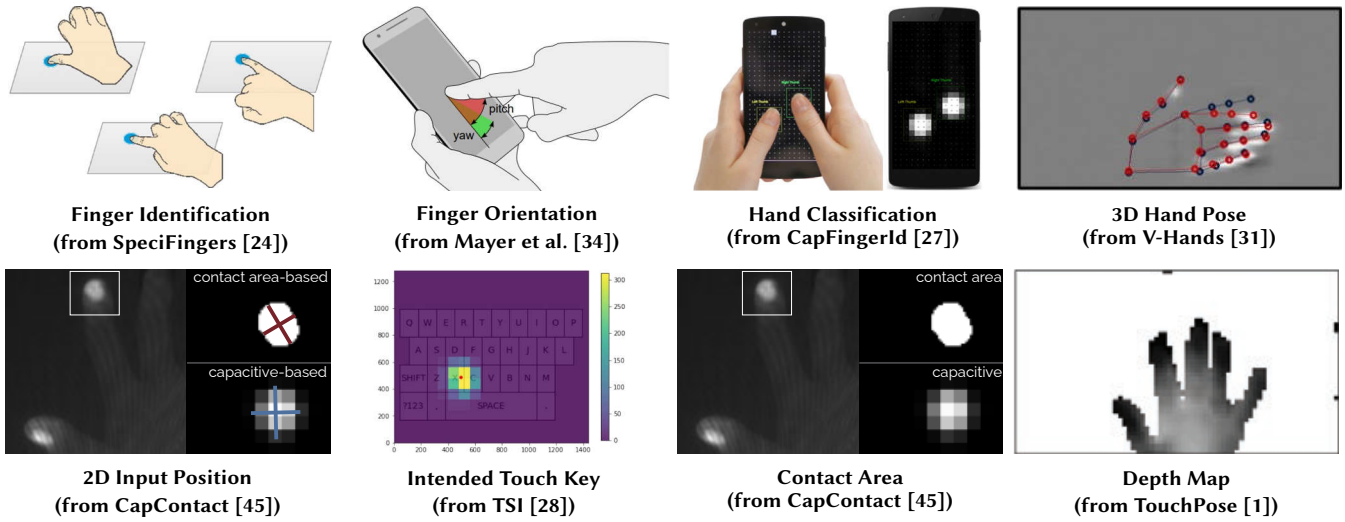
Figure 1: *CapFM* is a pretrained foundation model for downstream touch sensing tasks on capacitive images. Given a raw capacitive image as input, CapFM's encoder produces a shared representation that supports diverse downstream tasks using only lightweight task-specific heads. The same frozen backbone serves all tasks, making the model data-efficient for downstream learning, transferable across devices, and suitable for simultaneously powering multiple touch properties in interactive systems.

1 Introduction

Capacitive touchscreens are the dominant sensing surface in interactive computing. Most touch-sensitive devices, from watches and phones to tablets and large displays, rely on mutual-capacitance sensing to detect and track touch input [9]. The technology scales from small curved wearables to room-sized interactive tables, can be manufactured on flexible and transparent substrates at low cost, and senses not only direct contact but also a small band of hover above the surface [18]. These properties have made capacitive sensing the standard for input on virtually all consumer screen devices.

While commercial touch pipelines often reduce the sensor's output to mere 2D coordinates for tap and drag detection, prior research has demonstrated the substantial promise of processing capacitive images for rich interactive tasks. From commodity touchscreen devices, previous efforts have identified which finger produced a touch [24, 27] and estimated its pitch and yaw angle [34, 41] to inform interactive behavior. Other methods have recovered the full 3D hand skeleton from capacitive images [1, 31] to support whole-hand interaction and content manipulation. Super-resolution methods recover the precise contact area between finger and surface [35, 45], which reduces touch-location error by over 20% [45]. Capacitive images have also been used to detect touch force without a dedicated pressure sensor [4], to distinguish palm from finger contacts for inadvertent-touch rejection [25, 42], and to improve keyboard

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.
UIST '26, Detroit, MI, USA
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2856-3/26/11
https://doi.org/10.1145/3830398.3830703



Finger Identification (from SpeciFingers [24]) | Finger Orientation (from Mayer et al. [34]) | Hand Classification (from CapFingerId [27]) | 3D Hand Pose (from V-Hands [31])

2D Input Position (from CapContact [45]) | Intended Touch Key (from TSI [28]) | Contact Area (from CapContact [45]) | Depth Map (from TouchPose [1])

Figure 2: We evaluated CapFM on eight downstream tasks, where we connected a lightweight head to our frozen pretrained encoder and trained it on a subset of the labeled data from the respective dataset(s) as detailed in Table 1. Images taken and adapted from the referenced paper for task illustration purposes.

decoding on mobile devices [28]. These results demonstrate that capacitive sensing is an expressive modality, one that can transform commodity touchscreens into surfaces that understand not just where, but how the hand interacts with them.

However, each of these interactive capabilities has required its own model, its own architecture, and its own labeled dataset. While the field has advanced task by task, individual task-specific architectures typically do not transfer to other or new tasks. Every new capability has required starting from scratch with a dedicated data collection, a custom network design, and a full training cycle tied to a single sensing setup. This fragmentation has prevented shared knowledge from accumulating across tasks and methods over time.

Deepening the problem is the effort of curating labeled capacitive data. Ground-truth annotations require purpose-built capture rigs that integrate the touch sensor with secondary modalities such as depth cameras [1], table-sized optical sensing setups [45], or entire motion-capture systems [31, 34]. These setups are complex to build and calibrate, and they constrain the number of participants and interaction conditions that a study can cover. The resulting datasets are often on the order of tens of thousands of frames, and they are bound to a single sensor configuration. Differences in grid-line pitch, electrode layout, digitizer bit depth, and noise characteristics create domain gaps that prevent models trained on one device from generalizing to another [9].

In this paper, we introduce *CapFM*, a foundation model for capacitive touch sensing (Figure 1). We pretrain a single backbone on a combined corpus of unlabeled capacitive frames that we aggregated from seven public datasets with their own sensor configurations. Using our frozen pretrained backbone, we demonstrate that appending a lightweight task-specific head, trained on a small sample of a supervised dataset (e.g., 10%) produces results that surpass those reported in previous experiments with fully supervised networks geared towards a specific task—with a computational efficiency that is often lower than the respective method.

1.1 From task-specific models to a single pretrained backbone

CapFM implements a Masked Autoencoder [15] that operates on raw capacitive frames resampled to a canonical resolution to learn a shared representation. During pretraining, the model reconstructs randomly masked patches of the capacitive image. Through this process, the model learns the spatial structure of electric-field interactions [9] across devices without any task-specific labels. We aggregate over four million unlabeled frames from seven public datasets that span phone and tablet sensors, resolutions from 15×27 to 122×70, and frame rates from 5 to 124 fps. We normalize frames, which decouples the backbone from device-specific drive and sense line counts as well as from signal scales.

For downstream use, CapFM's pretrained encoder remains entirely frozen and produces a fixed set of spatial token embeddings from each input frame. To apply our model for a specific downstream task, a lightweight task head needs to be trained per task—on just a subset of a labeled dataset to already exceed baseline performance and without including the respective dataset during pretraining. The task head was either a two-layer MLP for global predictions or a small convolutional probe for spatial output.

We evaluated CapFM on eight downstream touch sensing tasks (Figure 2). Our results show that CapFM can replace the current practice of training a dedicated model for each task with a single pretrained backbone. Each task represents a distinct interactive capability previously addressed with a purpose-built model.

Finger Identification. Knowing which finger produces a touch enables per-finger shortcuts, hand-aware keyboard layouts, and implicit mode switching [12, 27, 32]. Combining CapFM with a classification head allows identifying the touching finger from its capacitive imprint, a task first introduced on phones by CapFingerID [27] and later scaled to tabletop surfaces by SpeciFingers [24].

Finger Orientation. The pitch and yaw angle of a finger at the moment of contact can parameterize continuous input such as for painting apps [1] or disambiguate intended targets on small displays [41, 48]. Coupling CapFM with a regression head affords estimating both angles from the shape of the capacitive imprint, a task addressed with a dedicated CNN by Mayer et al. [34].

Hand Classification. Distinguishing which hand the user touches with enables handedness-aware layouts, one-handed interaction modes, and per-hand role assignment in bimanual input [27, 32]. A classification head on CapFM can identify the touching hand from a single capacitive frame, a task that prior work has addressed as part of broader finger-identification pipelines [24, 27].

3D Hand Pose. Reconstructing the full 3D hand configuration from touch input enables gestural manipulation, context-aware interfaces, and remote collaboration on commodity touchscreens [1, 31]. A pose head following CapFM can recover 21-joint hand skeletons from a single capacitive frame, a problem first addressed by TouchPose on tablets [1] and extended to whiteboards by V-Hands [31].

2D Input Accuracy. Reducing the capacitive blob of a touch event to an accurate (x, y) coordinate as intended by the user enables UIs to be more reliably operated while packing more UI elements [3, 19]. Combining CapFM with a regression head helps predict accurate 2D Input Position following contact-aware processing [45]. In addition, inferring the Intended Touch Key rather than the registered coordinate reduces text-entry errors. Complementing CapFM with a classification head can predict the intended key from capacitive input captured during typing [28].

Contact Area. Resolving the contact area between finger and surface helps distinguish touching from hovering fingers and improves touch-input accuracy [45]. Adding to CapFM a convolutional head allows predicting binary contact masks from raw capacitive images, a cross-modal conversion task introduced by CapContact [45].

Depth Map. Estimating a per-pixel depth map of the hand above the touchscreen encodes finger width, volumetric shape, and other biometric characteristics, which complements the sparse joint skeleton and helps resolve ambiguities in single-touch poses [1, 31]. Coupling CapFM with a dense regression head allows reconstructing orthographic depth images from capacitive frames.

Taken together, three properties distinguish CapFM from the task-specific models that have addressed these capabilities individually. First, CapFM is *data-efficient*: self-supervised pretraining on unlabeled frames reduces the amount of labeled data that each downstream task requires. We show that as few as 10% of the labeled samples suffice to match or exceed fully supervised baselines. Second, CapFM *transfers across devices*: because our backbone learns the structure of capacitive fields rather than the characteristics of a specific sensor, it generalizes to sensor configurations that were not part of pretraining. Third, CapFM achieves these properties *with a single frozen backbone*: the same pretrained weights serve all eight tasks above. Only the lightweight head is trained per task. This also allows CapFM to *simultaneously power* all eight heads to jointly provide these estimates to a UI for applications.

1.2 Contributions

In this paper, we make the following contributions:

- CapFM, the first foundation model for capacitive touch sensing. Our model learns a shared representation from unlabeled capacitive frames and supports diverse touch-understanding tasks from a single frozen backbone with lightweight task heads.
- a dual-objective pretraining scheme that combines masked reconstruction with auxiliary contact segmentation. We derive the pseudo-contact masks for the contact objective from each frame itself, so pretraining requires no labels.
- a large-scale pretraining corpus of over four million unlabeled capacitive frames. We aggregated this corpus across seven public datasets that span different devices, sensor resolutions, and interaction contexts.
- an evaluation of CapFM on eight downstream tasks, replicating the respective paper's evaluation regimen according to its description. We show that models powered by CapFM match or outperform each task-specific baseline while requiring substantially fewer labeled examples and holding the respective dataset out from pretraining.

We will release our models, tests, and the unified data corpus with references to the source datasets for reproducibility.

2 Related work

CapFM is related to interaction techniques that leverage capacitive touch sensing, learning-based processing of capacitive images, and self-supervised representation learning.

2.1 Rich interaction from capacitive images

Capacitive sensing as an input modality beyond touch-location detection has a long history in HCI. DiamondTouch [6] and SmartSkin [39] first demonstrated that the drive and sense line matrix pattern supports touch separation, user differentiation, and shape recognition through image processing on the resulting 2D intensity maps. This approach was later translated to commodity touchscreen devices [22, 48], which commonly operate on the same principle. Grosse-Puppendahl et al. provided a survey on capacitive sensing configurations and applications across the field [9].

Over the past several decades, researchers have shown that raw capacitive images encode interaction properties well beyond input location. For finger orientation, AnglePose's particle filter over a geometric finger model tracked pitch and yaw from a low-resolution sensor array [41]. Goguey et al. measured naturally occurring pitch and roll ranges across all ten fingers during tap, drag, and flick actions to establish empirical baselines for which angular ranges are feasible as input channels [8].

The rise of deep learning methods in interactive computing opened up new opportunities for extracting richer information from capacitive touch images. For example, CNN-based methods reduced the angular error of finger-tilt estimation by operating directly on the capacitive image [34, 48]. TrackPose added temporal self-attention for jitter-free continuous estimation [16] and SwivelTouch classified discrete finger rotations as gesture primitives [29]. For finger identification, CapFingerID established CNN-based classification from capacitive images on phones [27], which SpeciFingers improved in robustness through spatio-temporal processing and

Table 1: Overview of the seven datasets of capacitive images and tasks we aggregated for CapFM's self-supervised pretraining. The combined corpus has over 4 M frames that span multiple device types, form factors, resolutions, and interaction modalities.

Dataset	Participants	Frames	Size ($W \times H$)	fps	Device & Placement	Primary Context
CapFingerID [27]	20	455,709	15×27	20	Phone (In-hand)	Finger ID
CapContact [45]	10	28,099	72×41	5	Tabletop (On desk)	Contact Mask
Mayer et al. [34]	33	1,155,656	15×27	20	Phone (In-hand)	Finger Pitch/Yaw Orientation
SpeciFingers [24]	20	1,786,891	122×70	100	Tabletop (On desk)	Finger ID
TouchPose [1]	10	65,374	72×41	5	Tabletop (On desk)	Depth, 3D Hand Pose
TSI [28]	16	43,735	39×18	124	Phone (In-hand)	Intended Keyboard Key
V-Hands [31]	16	735,364	71×41	15	Tabletop (On desk)	3D Hand Pose
Total & Range	**125**	**~4.28M**	**15×27 to 122×70**	**5–124**	**Mobile & Tabletop**	**Multi-Modal Spatial/Temporal Coverage**

error correction [24]. InfiniTouch applied CNN-based processing to a fully sensor-wrapped smartphone prototype with over 95% accuracy [26]. Together, these capabilities have enabled per-finger shortcuts [13], porous multi-layer interfaces [12], and systematic finger-role assignment across form factors [32].

A long-standing challenge has also been in identifying persons from touch events, for which prior work has proposed distinguishing two persons [14], providing users with a wearable [23, 36] or connecting their chair to the touch surface [6] to augment touch sensing, altering the touch sensing principle [21, 38], or augmenting devices with additional sensors (e.g., cameras [40] or motors [49]).

For hand pose, TouchPose recovered 21-joint 3D skeletons from a single capacitive frame using a multi-task CNN [1]. Choi et al. independently demonstrated 3D hand pose estimation on a commodity Samsung tablet using a custom kernel to expose raw capacitive data [5]. V-Hands extended this to bimanual tracking with a GRU-based temporal network [31].

For hand-part classification, PalmTouch distinguished palm from finger contacts at 99.5% accuracy for inadvertent-touch rejection [25], Schwarz et al. addressed palm rejection during stylus use through spatiotemporal touch features [42], and Matulic et al. generalized hand-contact-shape recognition to seven distinct postures on tabletops [33]. Zhang et al. instrumented a tablet bezel with capacitive sensing to detect arm orientations from bezel signals in combination with touch heatmaps [51].

For contact geometry, CapContact trained a cross-modal super-resolution GAN on paired capacitive frames and images from a frustrated total internal reflection (FTIR) setup for ground-truth contact masks at 8× the capacitive sensor resolution, which reduced touch-location error by over 20% [45]. Multi-frame super-resolution has further increased the accuracy of geometric metrics [35].

Contact shape efforts are related to researchers' broader efforts on modeling and improving touch-input accuracy. Previous work has identified the systematic offsets caused by finger pose in touch-input tasks [17, 19, 20]. Bayesian target-selection models [3] have shown that richer spatial information from the sensor directly improves input precision. For text entry specifically, TOAST enabled ten-finger eyes-free typing on a capacitive tabletop at over 44 WPM [44], TypeBoard used contact features to distinguish intentional keystrokes from resting touches at 98.9% accuracy [10], and ResType calibrated an invisible adaptive keyboard from detected resting-finger contacts [30]. PalmBoard leveraged pressure-image features for one-handed decoding without visual keyboard feedback [50]. Most recently, feeding raw capacitive heatmaps into keyboard decoders has been shown to reduce character error rates by over 21% on commodity phones [28].

Beyond touch input, interactive systems have leveraged capacitive sensing for hover and grip detection even before contact. For example, PreTouch leveraged self-capacitance sensing inside the touchscreen to sense farther and thus anticipate input events [18]. Researchers have also processed capacitive signals for software-based force sensing from contact-shape changes [4], biometric identification from body-part imprints [11, 22], and object recognition on capacitive fabric surfaces [47].

2.2 Learning from capacitive images

The capabilities above share a common trajectory: early systems relied on hand-crafted features and geometric models [41, 46], while recent work has shifted to CNNs trained end-to-end on raw capacitive images [1, 27, 34, 45]. This shift has improved accuracy substantially, but it has also introduced a structural limitation. Each model is trained in isolation on a single dataset collected with a specific sensor. Differences in electrode layout, grid-line pitch, digitizer bit depth, and noise characteristics create domain gaps that prevent transfer across hardware [9].

This fragmentation is compounded by a data bottleneck. Ground-truth labels require synchronized external setups such as FTIR rigs [45], depth cameras [1], or motion-capture systems [31]. These are expensive to build and constrain the diversity of participants and conditions a study can cover. As a result, capacitive datasets remain small, typically tens of thousands of frames, and no labeled corpus spans multiple devices.

2.3 Self-supervised pretraining

Self-supervised pretraining addresses data scarcity and domain fragmentation by learning representations from unlabeled data. Masked Autoencoders learn by reconstructing randomly masked patches of an input and produce backbones that transfer to downstream tasks with limited supervision [15]. Self-supervised pretraining on gaze data has been shown to reduce calibration requirements for eye tracking [2]. CapFM brings this approach to capacitive sensing by leveraging an aggregated dataset corpus for learning a shared representation that transfers across tasks and sensor configurations.

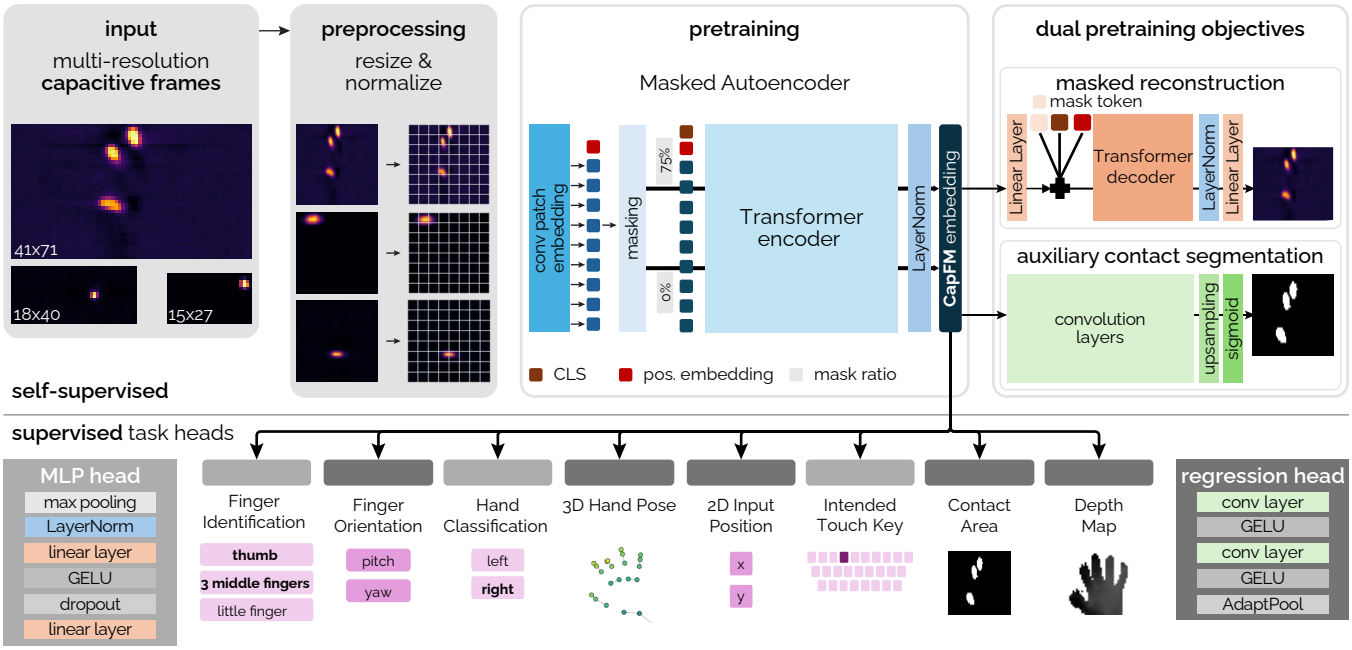


Figure 3: CapFM's architecture: self-supervised component of our model and supervised heads for downstream tasks.

3 CapFM Method

We designed CapFM as a unified, self-supervised foundation model to learn generalized representations of raw capacitive touch data across different devices and interaction contexts.

3.1 Datasets

To train a robust foundation model for capacitive touch and expose the model to diverse hand geometries, interaction styles, and hardware configurations, we aggregated seven publicly available datasets. These span data from mobile, tablet, and tabletop device form factors, varying resolutions, and diverse interaction modalities during capture. As shown in Table 1, our aggregated pretraining corpus comprises approximately 4.3 million capacitive frames from 125 unique participants.

Device form factors and sensor settings. The datasets cover mobile in-hand devices (*CapFingerID* [27], *TSI* [28], *Orientation* [34]), tablets (*TouchPose* [1], *CapContact* [45]), and tabletop sensors (*SpeciFingers* [24], *V-Hands* [31]). Phone sensors had drive and sense-line grids of 15×27 to 39×18 and sampling rates of 5–124 fps, whereas tablet and tabletop sensors had higher resolutions (71×41 to 122×70) and lower fps. This diversity introduces much variation in spatial resolution, temporal sampling, and noise characteristics. In turn, this challenges our model to learn robust and generalizable representations rather than overfitting to a single device.

Interaction modalities. The aggregated corpus comprises a rich array of interaction techniques and ground-truth labels. These include geometric tasks such as 3D pose estimation and orientation (pitch/yaw), classification tasks like finger identity, and intended key presses. Exposing the model to this variety in interaction types encourages the network to learn the underlying physics of capacitive fields during pretraining. This enables device-agnostic generalization across unseen sensors, users, and interaction styles.

3.2 Data Representation and Preprocessing

We represent each raw capacitive frame as a single-channel intensity map that captures the spatial distribution of electric field changes [9]. To register frames across different sensors into one coordinate system, we resample all frames to a canonical resolution of 64×64 using linear interpolation. This standardization compensates for differences in array size, aspect ratio, and electrode count while preserving the coarse spatial structure of the underlying signal.

Note that the purpose of this preprocessing step is canonicalization rather than removal of hardware differences. Across the datasets we aggregate, capacitive sensors operate in a broadly similar physical regime, with electrode pitches around 4 mm [35]. Because electrode pitch is broadly comparable, resampling to a common grid primarily standardizes array extent and sampling density rather than inventing fine spatial detail. This results in a shared spatial coordinate system across devices, while substantial variation remains in exact sensor layout, dynamic range, ADC resolution, grounding conditions, noise characteristics, and interaction context.

To further reduce inter-device discrepancies in signal scale, we apply per-frame min-max normalization and map each sample to $[0, 1]$. This encourages the model to focus on relative spatial structure rather than absolute signal magnitude.

Unlike standard Masked Autoencoders, we augment our pretraining with an auxiliary contact segmentation objective that encourages sensitivity to physically meaningful touch signals. Each input $\mathbf{x}$ is paired with a soft pseudo-contact mask $M \in \mathbb{R}^{B\times1\times64\times64}$ that provides explicit self-supervision for contact localization. We

derive these pseudo-contact masks as follows: Given an input frame $\mathbf{x}$, we construct

$$M = (1 - \lambda)\,\mathbf{S} + \lambda\,\mathbf{G}, \tag{1}$$

where $\mathbf{S} = \sigma\left(\frac{\mathrm{X}-q}{\tau}\right)$ is a saliency map based on an adaptive quantile threshold q, and $\mathbf{G}$ is the normalized Sobel gradient magnitude to capture the boundaries. The saliency term detects where contact likely exists and the gradient where it is likely to end. This combination captures both regions of strong signal and their spatial boundaries and produces approximations of contact extent under sensor noise.

3.3 Self-Supervised Model Architecture

Our model implements a dual-objective Masked Autoencoder (MAE) to learn hardware-agnostic representations of capacitive signals as shown in Figure 3. The model comprises a shared Vision Transformer (ViT) encoder and two self-supervised pretraining branches: a reconstruction decoder and an auxiliary contact segmentation head for dense pseudo-mask supervision [7]. This enables the backbone to capture both the global distribution of capacitive signals and the spatial boundaries of touch contacts. We obtained the selected parameters through hyperparameter optimization.

3.3.1 Input Tokenization. Each capacitive frame $x \in \mathbb{R}^{B\times1\times64\times64}$ is divided into non-overlapping 8×8 patches via a 2D convolution (kernel=8, stride=8), yielding $L = 64$ patch tokens. We select a resolution of 64×64 as a balance between preserving spatial fidelity and maintaining computational efficiency. This resolution aligns with the 8×8 patch size used for tokenization, yielding a fixed grid of 64 tokens. These tokens are projected into a D-dimensional latent space and combined with fixed 2D sine-cosine positional embeddings with $D = 384$, providing stable spatial context across varying sensor layouts and resolutions. This patch-based tokenization balances spatial granularity with computational efficiency, preserving enough local structure for downstream tasks.

3.3.2 Masked Reconstruction Branch. The reconstruction branch follows the standard MAE framework with a high masking ratio of $r = 0.75$ to encourage global reasoning over the capacitive field, compelling the encoder to infer missing patches from limited visible context. The encoder consists of a six-layer Transformer with eight attention heads and an MLP ratio of 4.0, while the lightweight decoder uses four layers with embedding dimension 128 and four heads. The branch is optimized with Mean Squared Error (MSE) loss ($\mathcal{L}_{recon}$) computed only over the masked patches.

3.3.3 Auxiliary Contact Segmentation Branch. To ground the model in physical interaction semantics, we utilize an auxiliary contact head that operates on unmasked encoder outputs (where mask ratio = 0). The resulting patch tokens are reshaped into a spatial feature map and passed through a convolutional decoder (two 3×3 layers and one 1×1 layer). The output is bilinearly upsampled to the original 64×64 resolution for dense supervision against the pseudo-contact masks. The contact head is intentionally lightweight to ensure that the burden of representation learning remains in the shared encoder, preventing the auxiliary branch from overfitting to dataset-specific contact patterns.

3.3.4 Dual-Objective Optimization. The pretraining loss combines the reconstruction and auxiliary contact segmentation objectives:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{recon}} + \lambda_{\text{contact}} \cdot \Phi(e) \cdot \mathcal{L}_{\text{contact}}, \tag{2}$$

where $\lambda_{\text{contact}} = 0.15$ and $\Phi(e) \in [0, 1]$ is a linear warmup schedule over training epochs e.

The contact branch employs a hybrid Focal-Dice loss:

$$\mathcal{L}_{\text{contact}} = (1 - \lambda_{\text{Dice}})\,\mathcal{L}_{\text{Focal}} + \lambda_{\text{Dice}}\,\mathcal{L}_{\text{Dice}}, \tag{3}$$

with $\lambda_{\text{Dice}} = 0.45$. This dual-objective formulation first prioritizes stable reconstruction of the capacitive field before gradually introducing contact supervision. As a result, localization improves and representations become more robust.

3.4 Supervised Task Heads

To evaluate the representational quality of the learned backbone, we attach lightweight task-specific heads while keeping the MAE encoder frozen. The encoder operates without masking and produces a fixed grid of spatial tokens, which serve as a shared representation for all downstream tasks.

Classification. For classification, we aggregate spatial tokens via element-wise max pooling, emphasizing high-activation regions corresponding to sparse touch events. The pooled feature is passed through a two-layer MLP with LayerNorm, a 256-dimensional hidden layer, GELU activation, dropout (0.1), and a final linear layer producing class logits. The tasks include Finger Identification, specifically differentiating between thumb, little finger and the remaining three fingers (coarse-3), Hand Classification, and Intended Touch Key all using the cross-entropy loss.

Regression. For regression tasks, we preserve spatial structure by reshaping tokens into a $384 \times 8 \times 8$ feature map and applying a lightweight convolutional probe consisting of a 1×1 convolution, GELU, a 3×3 convolution, GELU, global average pooling, and a final linear projection. We consider four regression settings:

(1) *Finger Orientation:* Predicts the two-dimensional vector $\mathbf{y} = [\theta_{\text{pitch}}, \psi_{\text{yaw}}]^\top$ using mean squared error.
(2) *3D Hand Pose:* Predicts a 63-dimensional vector of (x, y, z) coordinates for the 21 hand joints using Smooth L1 loss.
(3) *Contact Area:* Predicts dense contact on the 8×8 token grid. Ground-truth maps ($8\times$ the capacitive frame resolution [45]) are downsampled to match the original resolution, and predictions are trained with mean squared error after a sigmoid activation. We upsample outputs with bilinear interpolation merely for evaluating and comparing dense metrics with prior work.
(4) *2D Input Position:* Predicts the two-dimensional touch location $\mathbf{y} = [c_x, c_y]^\top$ in image coordinates using mean squared error.

4 Evaluation

We evaluated CapFM across a diverse set of touch sensing tasks on capacitive touchscreens. Our evaluation tested three core hypotheses: (H1) A unified foundation model outperforms specialized, task-specific architectures. (H2) Self-supervised pretraining yields superior representations compared to random initialization. (H3) These representations enable strong data efficiency and require a fraction of labeled data to achieve competitive performance.

4.1 Implementation Details

Pretraining. The backbone is trained using the dual-objective MAE framework with AdamW (learning rate 3×10^{-4}, weight decay 0.05, batch size 64). To stabilize early gradients, we employed a linear warmup for the first 5 epochs, followed by a cosine annealing learning rate schedule. Pretraining used a single-process CUDA loop, which focused on optimizing the dual-objective loss across the unified 7-dataset corpus. Early stopping happens based on reconstruction loss convergence, resulting in an effective training duration of 36 epochs (~15 hours on an NVIDIA RTX 4090 GPU).

Downstream Training. For each downstream task, we keep the MAE encoder frozen and optimize solely the task-specific heads. We use AdamW with a learning rate of 10^{-3}, weight decay 0.01, and a batch size of 256. Training is performed for 40 epochs across all tasks. Loss functions are selected based on task type: cross-entropy for classification, MSE for orientation and contact regression, and Smooth L1 loss for 3D pose estimation. This setup is consistent across all tasks to ensure that performance differences reflect representation quality rather than task-specific optimization. Gradients are computed only for the task-specific heads, ensuring that the shared backbone remains unchanged.

Data Splits and Evaluation Protocol. All datasets are split at the participant level to ensure generalization to unseen users. Specifically, 10% of participants are held out per dataset as a fixed test set and are never observed during training or validation. The remaining 90% form the training pool, from which 10% are reserved for validation to ensure no subject overlap across splits.

4.2 Model Complexity and Inference Efficiency

Our MAE instantiates the architecture with a total of 12.07M parameters, including 11.56M in the encoder–decoder backbone and 0.52M in the auxiliary contact head (12.04M trainable). We provide a comparison of parameter counts and computational complexity with prior capacitive sensing architectures in Table 11.

To evaluate practical deployment, we also implemented the encoder on a HUAWEI HBN-AL10 smartphone (Kirin 9010 SoC). Using native CPU inference on 40×18 capacitive images while producing all eight downstream predictions, the complete pipeline achieved 10.36 fps, including preprocessing. The deployed encoder consists of 10.70M parameters (0.692 GMACs / 1.384 GFLOPs), indicating that CapFM is lightweight enough for real-time execution on contemporary mobile hardware.

These results demonstrate CapFM's real-time performance at 800+ fps on a GPU, ~90 fps on a desktop CPU, and interactive performance on commodity mobile hardware, while maintaining a single shared backbone across downstream tasks.

4.3 Baseline Implementation and Modification

For systematic comparison, we evaluate CapFM against architectures and results reported in prior work on downstream tasks. Where original implementations were publicly available (CapFingerID, CapContact, TouchPose, Mayer et al., V-Hands, SpeciFingers), we retrain baseline models from scratch using our unified data splits. This ensures that observed differences are attributable to model design and pretraining, rather than discrepancies in evaluation protocol. For Google's TSI dataset, where code was not available, we compare against the reported metrics from the original publication.

Our evaluation further adopts a hardware-agnostic, spatial-only formulation. All models operate on a single capacitive frame ($T = 1$). For datasets that originally relied on temporal modeling (e.g., LSTMs or 3D CNNs in SpeciFingers and V-Hands), we adapt these architectures to remove temporal dependencies. This enforces a consistent evaluation setting focused purely on spatial representation quality and avoids confounding factors that may stem from different sensor frame rates across datasets.

4.4 Evaluation Protocols

We evaluate the model under three complementary protocols:

Protocol P1: Multi-task benchmark. All tasks are evaluated using a shared 90/10 participant-level split. This protocol assesses the ability of a single frozen backbone to serve as a general-purpose feature extractor across heterogeneous datasets.

Protocol P2: Data Efficiency and Ablation. We compare the frozen backbone against a randomly initialized version of the same architecture. We also allow for finetuning in the randomly initialized architecture. We also compare the impact of the max pooling by evaluating our model using mean pooling instead. To show the impact of our auxiliary contact branch, we evaluate a plain MAE trained on the reconstruction objective alone. To assess sample efficiency, we evaluate the model using limited subsets of the downstream labels: 10%, 25%, and 50% fractions of the training data. This isolates the impact of representation quality on sample efficiency.

Protocol P3: Original Paper Evaluation Setting and Generalization Benchmark. We evaluate under the original data splits used in prior work (e.g., Leave-One-Subject-Out) for direct comparison to published results. We also consider two pretraining regimes: (i) full pretraining on all datasets (including just the training portion of the downstream task dataset) and (ii) excluded pretraining, where the target dataset is omitted during pretraining to assess cross-dataset generalization.

4.5 Evaluation Metrics

We adopt evaluation metrics consistent with prior work to enable direct comparison across tasks. For classification tasks (Finger Identification, Hand Classification), we report classification accuracy. For regression tasks, we report standard error metrics. For Finger Orientation estimation, we evaluate RMSE and MAE in degrees for pitch and yaw [34]. For 3D Hand Pose estimation, we follow established protocols [1, 31] and report MSE and 3D End-Point Error (EPE) in millimeters. We further decompose EPE into planar (EPE_{xy}) and depth (EPE_z) components to provide finer-grained analysis. EPE (All / Visible) refers to end-point error over all points vs. only visible points. EPE (XY / Z) specifies errors along horizontal/vertical vs. depth dimensions.

For 2D Input Position and Depth Map, errors are reported in millimeters. For Intended Touch Key on TSI [28], we follow Lertvittayakumjorn et al. and report Character Error Rate (CER), which reflects real-world text entry performance. For Contact Area estimation, we report pixel-level intersection over union (IoU), precision, recall, and F1 score, along with geometric measures such as area

Table 2: Accuracy for Finger Identification. *The published SpeciFingers result uses a temporal sequence as input. †Whether training split was included in pretraining.

Protocol	Setting	CapFingerID	SpeciFingers
P1	CapFM frozen	**71.23%**	**52.99%**
Multi-task	CFID [27]/SF [24]	65.94%	50.91%
P2	10% labels	69.83%	45.99%
Efficiency	25% labels	70.43%	48.04%
	50% labels	71.20%	50.12%
P2	Random init	58.70%	48.52%
Ablation	⌞ + finetune	62.57%	51.51%
	Mean pooling	63.11%	53.73%
	Plain MAE	69.55%	51.27%
P3	CFID/SF	**72.19%**	59.21%–**82.12%***
CFID/SF	CapFM		
protocol	⌞ CFID/SF excluded†	65.46%	65.65%
	⌞ split included†	65.81%	**77.49%**

Table 3: Finger Orientation RMSE/MAE on Mayer et al. [34]. †Whether training split was included in pretraining.

Protocol	Setting	Overall		Pitch		Yaw	
		RMSE	MAE	RMSE	MAE	RMSE	MAE
P1	CapFM frozen	**32.64**	**15.99**	**13.20**	**10.15**	**29.85**	**21.84**
Multi-task	Mayer et al. [34]	33.70	25.80	15.03	12.26	30.23	22.72
P2	10% labels	32.86	16.22	13.44	10.41	29.99	22.04
Efficiency	25% labels	2.48	16.04	13.38	10.34	29.60	21.74
	50% labels	32.21	15.95	13.29	10.31	29.34	21.60
P2	Random init	43.74	22.30	14.70	11.80	41.20	32.90
Ablation	⌞ + finetune	39.71	18.60	14.36	11.02	37.02	26.36
	Mean pooling	42.35	21.78	14.54	11.61	39.78	31.95
	Plain MAE	30.82	15.16	13.22	10.30	27.84	20.02
P3	Mayer et al.	**28.24**	20.47	13.33	10.47	**24.90**	**18.34**
Mayer	CapFM						
et al.	⌞ excluded†	30.26	15.18	13.49	10.73	27.09	19.63
prot.	⌞ included†	29.35	**14.74**	**13.12**	**10.43**	26.25	19.03

size difference and center distance. These metrics quantify how accurately the predicted capacitive contact maps match the ground truth from optical FTIR-based masks [45].

5 Results

We first provide an overview of the results by task for Finger Identification (Table 2), Finger Orientation (Table 3), Hand Classification (Table 4), 3D Hand Pose (Table 5), 2D Input Position (Table 6), Intended Touch Key (Table 7), Contact Area (Table 8), and Depth Map (Table 9). All tables report the task-specific metrics introduced in Section 4.5 and organize results by evaluation protocol.

5.1 Overall Multi-Task Performance (P1)

Under the shared multi-task setting, our single frozen CapFM backbone transfers well across all eight tasks and is competitive with, or better than, the corresponding task-specific models. The largest

Table 4: Hand Classification Accuracy on CapFingerID [27].

Protocol	Setting	Accuracy
P1	CapFM frozen	64.79%
Multi-task	CapFingerID [27]	**64.84%**
P2	10% labels	64.11%
Efficiency	25% labels	64.11%
	50% labels	64.56%
P2	Random init	51.40%
Ablation	⌞ + finetune	48.60%
	Mean pooling	54.10%
	Plain MAE	62.30%
P3	CapFingerID	59.27%
CapFingerID	CapFM	
protocol	⌞ CapFingerID data excluded	61.38%
	⌞ CapFingerID training split included	**64.58%**

Table 5: 3D Hand Pose MSE and endpoint errors on V-Hands [31]. †Whether training split was included in pretraining. *Published result with temporal-sequence input.

Prot.	Setting	MSE	EPE		EPE XY		EPE Z	
		($\times 10^{-2}$)	All	Vis.	All	Vis.	All	Vis.
P1	CapFM frozen	**1.31**	**33.5**	**31.4**	**31.8**	**30.0**	**7.3**	**6.1**
	V-Hands [31]	1.43	104.28	103.79	96.64	93.27	33.14	34.97
P2	10% labels	.69	30.6	28.4	28.9	27.0	6.9	6.0
	25% labels	.14	30.2	28.3	28.4	26.8	7.0	6.2
	50% labels	1.13	31.5	29.5	29.8	28.2	6.9	5.9
P2	Random init	.18	63.4	63.5	61.5	62.0	10.0	9.0
Abl.	⌞ + finetune	.11	51.9	50.2	50.1	48.8	9.0	8.0
	Mean pooling	1.48	44.7	42.8	41.5	39.9	11.9	11.1
	Plain MAE	.50	34.1	31.6	32.1	30.1	7.9	6.6
P3	V-Hands	1.43	104.3	103.8	94.6	93.3	33.1	34.9
	sequence-based*		**11.8**	**9.7**	**9.0**	**7.5**	**6.0**	**4.8**
V-H.	CapFM							
prot.	⌞ V-Hands excl.†	0.05	31.3	28.9	29.5	27.5	7.2	6.1
	⌞ train split incl.†	**0.04**	**29.9**	**27.7**	**28.1**	**26.1**	**7.2**	**6.5**

gains appear on dense geometric tasks. For Contact Area, CapFM improves IoU from .701 to .725 and reduces area error from 6.93 to 3.11. For Depth Map, it lowers depth error from 22.21 mm to 18.63 mm. We also observe gains for Finger Orientation, where overall RMSE decreases from 33.70° to 32.64°, and for 3D Hand Pose, where all endpoint-error measures improve substantially over the single frame version of the original task-specific model.

Classification results are also better, though with smaller gains. For Finger Identification, CapFM improves over the original models on both CapFingerID and SpeciFingers. For Intended Touch Key, character error rate drops from 6.32% to 3.59%, which shows that the shared pretrained representation also benefits discrete target prediction. By contrast, Hand Classification remains nearly unchanged (64.79% vs. 64.84%), which suggests that this task benefits less from pretraining than the more spatially structured tasks do.

Table 6: 2D Input Position error Δx, Δy in mm on CapContact [45]. †Whether training split was included in pretraining.

Protocol	Setting	Δx	Δy	d_{center}
P1	CapFM frozen	-0.20	**0.31**	**6.04**
Multi-task	CapContact [45]	**0.05**	1.22	6.53
P2	10% labels	0.60	−3.05	4.37
Efficiency	25% labels	−1.23	−0.46	4.77
	50% labels	−0.75	0.20	3.65
P2	Random init	−2.33	0.15	49.86
Ablation	⌞ + finetune	−0.32	0.59	4.80
	Mean pooling	−1.02	1.39	7.03
	Plain MAE	1.01	0.25	12.59
P3	CapContact	**−0.05**	1.22	**1.65**
CapContact	CapFM			
protocol	⌞ CapContact data excluded†	−0.92	−1.22	3.18
	⌞ training split included†	−0.87	**−0.30**	1.69

Table 7: Intended Touch Key character error rate on TSI [28].

Protocol	Setting	CER
P1	CapFM frozen	**3.59%**
Multi-task	TSI [28]	6.32%
P2	10% labels	5.28%
Efficiency	25% labels	4.31%
	50% labels	3.69%
P2	Random init	13.30%
Ablation	⌞ + finetune	4.48%
	Mean pooling	4.98%
	Plain MAE	3.69%
P3	TSI	6.32%
TSI	CapFM	
protocol	⌞ TSI data excluded from pretraining	6.28%
	⌞ TSI training split included	**3.69%**

Table 8: Contact Area errors on CapContact [45]. †Whether training split was included in pretraining.

Prot.	Setting	IoU	MSE	F1	Prec.	ΔA
P1	CapFM frozen	**.725**	**.001**	**.833**	**.865**	**3.11**
Multi-task	CapContact [45]	.701	.008	.819	.807	6.93
P2	10% labels	.649	.003	.767	772	4.18
	25% labels	.662	.001	.776	.765	8.58
	50% labels	.690	.001	.798	.781	10.33
P2	Random init	.580	.003	.669	.757	21.14
Abl.	⌞ + finetune	.747	.001	.829	.858	-6.32
	Mean pooling	.704	.001	.793	.822	5.55
	Plain MAE	.384	.003	.473	.585	-33.06
P3	CapContact	**.67**–.63	—	**.98**	**.96**	**2.41**
CapContact	CapFM					
protocol	⌞ data excluded†	.554	.001	.791	.817	5.38
	⌞ split included†	.636	.001	.904	.904	-2.47

Table 9: Depth Map per-pixel error on TouchPose [1].

Protocol	Setting	depth error (mm)
P1	CapFM frozen	**18.63**
Multi-task	TouchPose [1]	22.21
P2	10% labels	24.83
Efficiency	25% labels	20.93
	50% labels	18.94
P2	Random init	34.99
Ablation	⌞ + finetune	56.18
	Mean pooling	17.34
	Plain MAE	21.56
P3	TouchPose	22.27
TouchPose	CapFM	
protocol	⌞TouchPose data excluded	24.92
	⌞ TouchPose split included	**18.81**

The results also reveal where transfer remains limited. For 2D Input Position, CapFM improves Δy and center-distance error, but not Δx, where the original model remains slightly better. Similarly, while Finger Orientation improves overall, yaw remains clearly more difficult than pitch in the frozen backbone setting.

5.2 Data Efficiency (P2)

CapFM remains effective even when only a small fraction of downstream labels is available. Across tasks, performance is convincing with 10% of the training data and often changes only modestly as more labels are added.

This pattern is clear for the classification tasks. For Finger Identification, accuracy increases steadily with more labels on both CapFingerID and SpeciFingers. For Hand Classification, performance is nearly unchanged across 10%, 25%, and 50% of labels. For Intended Touch Key, character error rate improves consistently as more labels are introduced, dropping from 5.28% at 10% of the data to 3.69% at 50%.

Regression tasks show a similar trend, although the magnitude of the gain depends on the task. For Finger Orientation, errors decrease gradually with more data, and both pitch and yaw remain stable across label fractions. For Depth Map, depth error falls from 24.83 mm at 10% to 18.94 mm at 50%. For 3D Hand Pose, the endpoint errors remain in a narrow range across all three label fractions.

Dense prediction on contact areas is particularly stable. For Contact Area, IoU, F1, precision, and geometric errors only vary slightly from 10% to 50% of the labels. For 2D Input Position, the errors vary somewhat more across label fractions, while center-distance error remains low throughout.

5.3 CapFM Ablation (P2)

Comparing random initialization with and without finetuning, mean pooling, and a plain MAE without the auxiliary contact branch, we see that random initialization is usually the weakest setting. Finetuning improves over this baseline but does not recover the performance of the pretrained model in most cases. Plain MAE is the strongest ablation on most tasks and occasionally beats

the full model, as on Finger Orientation. The largest differences appear on structured geometric tasks (3D Hand Pose, Finger Orientation, Depth Map), where random initialization increases error markedly and finetuning only partly closes the gap. The differences are smaller for classification tasks. Contact geometry is the clear exception; here, plain MAE drops to .384 IoU on Contact Area, falls below random initialization, and doubles the center distance on 2D Input Position from 6.04 to 12.59. The auxiliary contact branch thus gives CapFM its largest advantage on the tasks that have been central to capacitive touch sensing as the dominant input modality: resolving where and how a finger contacts the surface.

5.4 Original Evaluation Protocol (P3)

In P3, we follow the original evaluation setting of each paper. Here, CapFM remains competitive with or better than the reported task-specific baselines on most tasks. When the training split of the downstream dataset is included during pretraining, performance improves over the original papers for Hand Classification (64.58% vs. 59.27%), Intended Touch Key (3.69% vs. 6.32% CER), and Depth Map (18.81 mm vs. 22.27 mm). It also improves the overall metric for Finger Orientation, changing RMSE/MAE from 28.24°/20.47° to 29.35°/14.74°. For 3D Hand Pose, the included setting reduces EPE from 104.3 mm to 29.9 mm relative to the original non-temporal baseline, although it remains above the sequence-based result. For Finger Identification, SpeciFingers improves over the non-temporal baseline, while CapFingerID remains below the original paper result. For 2D Input Position, Δy improves relative to the original protocol.

When the downstream dataset is excluded during pretraining, performance generally decreases but remains competitive on several tasks. The excluded model still outperforms the original baselines on Hand Classification and the Finger Orientation metric, and it remains close to the original result on Intended Touch Key. Exclusion leads to clearer drops on Finger Identification, Contact Area, Depth Map, and 2D Input Position, and it increases the gap to the strongest results on 3D Hand Pose.

6 Discussion

Overall, the results show that a single frozen encoder can support a broad range of capacitive sensing tasks with competitive performance across all evaluation protocols. Under P1, the same pretrained backbone transfers across classification, regression, pose estimation, segmentation, and input prediction, which indicates that these tasks share structure that a unified representation can capture. One encoder remains effective across such different outputs, which shows that capacitive sensing does not need to be modeled as a collection of fully separate learning problems.

The results also clarify what CapFM's shared representation captures well. Strong transfer on tasks such as contact geometry, depth, orientation, and hand pose suggests that the encoder preserves meaningful spatial structure in the capacitive field. At the same time, transfer is not uniform, since gains are smaller on some classification tasks and yaw remains harder than pitch in Finger Orientation. A shared encoder therefore appears most effective when downstream performance depends on recovering structured properties of the signal, while some tasks may still benefit from additional task-specific refinement.

The results also confirm that little labeled data is needed to train CapFM for downstream tasks. Performance is already convincing with only 10% of the downstream labels (P2). Many tasks improve only modestly as more labels are added. A substantial part of the useful signal has thus already been absorbed during pretraining. For capacitive sensing, this matters because many labels depend on specialized capture setups such as FTIR systems or motion tracking and are therefore costly to obtain.

Our ablation study indicates that the gains come from representation learning rather than architecture alone. Random initialization usually performs worst. Finetuning from random initialization does not recover the full model on most tasks. Plain MAE often comes close on the classification and pose tasks, but it falls well behind on Contact Area and 2D Input Position. This suggests that masked reconstruction provides a strong basis, while the auxiliary contact segmentation objective delivers the largest gains on dense contact geometry. Since contact location and extent are the primary quantities a capacitive sensor is asked to recover, this label-free objective carries the tasks that arguably may matter most as the most fundamental input decoding tasks on today's touch devices.

In terms of generalization (P3), performance is strongest when the downstream training split is represented during pretraining. However, the excluded setting remains competitive on several tasks and still exceeds original baselines in some cases. This suggests that the encoder captures features that transfer across datasets, devices, and sensing conditions rather than only memorizing dataset-specific patterns. At the same time, the drop under exclusion shows that pretraining breadth still matters and that some tasks remain sensitive to the target distribution.

A shared backbone also pays off at deployment. CapFM serves all eight heads from a single frozen encoder and produces every prediction in the same pass. The full pipeline runs at 10.36 fps on a commodity smartphone. Our results thus suggest that systems that today ship a separate model per capability can consolidate them into a single perceptual layer with lightweight heads.

Limitations and Future Work

While CapFM performs strongly across the task set, the gains are not uniform. Improvements are smaller for fine-grained classification tasks such as Hand Classification and for yaw in Finger Orientation. This suggests that some tasks may still benefit from limited task-specific refinement beyond a shared frozen backbone.

Our model is also limited to single-frame inputs. Prior work and our P3 results indicate that temporal models remain stronger for tasks such as 3D Hand Pose, which makes temporal modeling a promising step for future work. The evaluated datasets further cover a relatively narrow range of sensing conditions. They focus mainly on finger input on planar capacitive surfaces, so the current model is not yet validated for tangibles [35], temporal gestures, or curved and flexible sensors [37]. More broadly, as a foundation model CapFM is domain-specific in capacitive touch sensing. Future work could expand the pretraining corpus, incorporate temporal modeling, extend the backbone to additional sensor forms and multimodal modeling, and study lightweight adaptation mechanisms that preserve a shared representation.

7 Applications

We prototyped a series of UI widgets that show CapFM's downstream performance for interaction. We trained all eight task heads to simultaneously estimate all eight tasks above.

We streamed raw capacitive images from a Huawei Pura 70 Pro+ phone (18 × 40 pixels, 100 fps) to a PC to run inference on CapFM's encoder and the eight heads. Neither CapFM nor the task heads have seen data from this particular phone in training. Our demos thus show that our model transfers to another sensor and digitizer.

Our video figure shows our demos during interaction: First, in a painting app, fingers are assigned to specific colors for painting strokes, where finger angle simultaneously determines the width and orientation of the brush. Users can erase paint by scrubbing with their opposite hand. Second, a 3D viewer renders objects whose rotation is controlled by altering one's finger pose, while hand pose controls the camera pose. Third, our typing application packs many keys into a small area and shows side-by-side input detection upon touch following the phone's reported (x,y) coordinate, a centroid-based estimate [45], and estimating input key selection directly [28].

8 Conclusion

We have introduced CapFM, a foundation model that replaces separate task-specific capacitive sensing pipelines with a single shared representation. We integrated more than four million capacitive frames from existing public datasets into a self-supervised training corpus. We pretrained CapFM on two label-free objectives: masked reconstruction and auxiliary contact segmentation, for which we derive pseudo-contact masks directly from the input frames. The resulting frozen encoder supports eight downstream tasks, from classification and regression to dense prediction and 3D hand-pose estimation, through lightweight task-specific heads.

Across these tasks, CapFM matches or outperforms specialized baselines under a shared evaluation protocol, remains effective with only a fraction of the available labels, and transfers across datasets and to sensing hardware not seen during pretraining. It achieves this from one reusable encoder rather than eight independently trained feature extractors. A single encoder forward pass drives all task heads at once, so CapFM estimates multiple touch properties in parallel and shares pretraining, model storage, and encoder computation across them. Individual heads can be added, replaced, or retrained without rebuilding the shared encoder.

More broadly, CapFM establishes a common perceptual layer for capacitive touch sensing. It consolidates knowledge from datasets collected for different tasks and devices into one reusable representation, so advances that would otherwise remain isolated in task-specific pipelines can accumulate in a shared model. Future systems can reuse this pretrained perceptual layer and extend it with task heads as needed. We believe that this can make capacitive interfaces easier to develop, adapt, and scale across current and future hardware and applications.

References

[1] Karan Ahuja, Paul Streli, and Christian Holz. 2021. TouchPose: Hand Pose Prediction, Depth Estimation, and Touch Classification from Capacitive Images. In *The 34th Annual ACM Symposium on User Interface Software and Technology (UIST '21)*. ACM, New York, NY, USA, 997–1009. doi:10.1145/3472749.3474801

[2] Louise Gillian C. Bautista and Prospero C. Naval Jr. 2020. GazeMAE: General Representations of Eye Movements using a Micro-Macro Autoencoder. arXiv:2009.02437 [cs.CV] https://arxiv.org/abs/2009.02437

[3] Xiaojun Bi and Shumin Zhai. 2013. Bayesian Touch: A Statistical Criterion of Target Selection with Finger Touch. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology (UIST '13)*. ACM, New York, NY, USA, 51–60. doi:10.1145/2501988.2502058

[4] Tobias Boceck, Sascha Sprott, Huy Viet Le, and Sven Mayer. 2019. Force Touch Detection on Capacitive Sensors Using Deep Neural Networks. In *Proceedings of the 21st International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '19)*. ACM, New York, NY, USA, Article 42, 6 pages. doi:10.1145/3338286.3344389

[5] Frederick Choi, Sven Mayer, and Chris Harrison. 2021. 3D Hand Pose Estimation on Conventional Capacitive Touchscreens. In *Proceedings of the 23rd International Conference on Mobile Human-Computer Interaction (MobileHCI '21)*. ACM, New York, NY, USA, Article 3, 13 pages. doi:10.1145/3447526.3472045

[6] Paul Dietz and Darren Leigh. 2001. DiamondTouch: a multi-user touch technology. In *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology* (Orlando, Florida) *(UIST '01)*. ACM, New York, NY, USA, 219–226. doi:10.1145/502348.502389

[7] 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. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. arXiv:2010.11929 [cs.CV] https://arxiv.org/abs/2010.11929

[8] Alix Goguey, Géry Casiez, Daniel Vogel, and Carl Gutwin. 2018. Characterizing Finger Pitch and Roll Orientation During Atomic Touch Actions. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, 1–12. doi:10.1145/3173574.3174163

[9] Tobias Grosse-Puppendahl, Christian Holz, Gabe Cohn, Raphael Wimmer, Oskar Bechtold, Steve Hodges, Matthew S. Reynolds, and Joshua R. Smith. 2017. Finding Common Ground: A Survey of Capacitive Sensing in Human-Computer Interaction. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 23 pages. doi:10.1145/3025453.3025808

[10] Yizheng Gu, Chun Yu, Xuanzhong Chen, Zhuojun Li, and Yuanchun Shi. 2021. TypeBoard: Identifying Unintentional Touch on Pressure-Sensitive Touchscreen Keyboards. In *The 34th Annual ACM Symposium on User Interface Software and Technology (UIST '21)*. ACM, New York, NY, USA, 14 pages. doi:10.1145/3472749.3474770

[11] Anhong Guo, Robert Xiao, and Chris Harrison. 2015. CapAuth: Identifying and Differentiating User Handprints on Commodity Capacitive Touchscreens. In *Proceedings of the 2015 International Conference on Interactive Tabletops and Surfaces (ITS '15)*. ACM, New York, NY, USA, 59–62. doi:10.1145/2817721.2817722

[12] Aakar Gupta, Muhammed Anwar, and Ravin Balakrishnan. 2016. Porous Interfaces for Small Screen Multitasking using Finger Identification. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16)*. ACM, New York, NY, USA, 145–156. doi:10.1145/2984511.2984557

[13] Aakar Gupta and Ravin Balakrishnan. 2016. DualKey: Miniature Screen Text Entry via Finger Identification. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 59–70. doi:10.1145/2858036.2858052

[14] Chris Harrison, Munehiko Sato, and Ivan Poupyrev. 2012. Capacitive Fingerprinting: Exploring User Differentiation by Sensing Electrical Properties of the Human Body. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology (UIST '12)*. ACM, New York, NY, USA, 8 pages. doi:10.1145/2380116.2380183

[15] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. 2021. Masked Autoencoders Are Scalable Vision Learners. arXiv:2111.06377 [cs.CV] https://arxiv.org/abs/2111.06377

[16] Ke He, Chentao Li, Yongjie Duan, Jianjiang Feng, and Jie Zhou. 2023. TrackPose: Towards Stable and User Adaptive Finger Pose Estimation on Capacitive Touchscreens. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 7, 4, Article 161 (2023), 22 pages. doi:10.1145/3631459

[17] Niels Henze, Enrico Rukzio, and Susanne Boll. 2011. 100,000,000 taps: analysis and improvement of touch performance in the large. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*. ACM, New York, NY, USA, 133–142.

[18] Ken Hinckley, Seongkook Heo, Michel Pahud, Christian Holz, Hrvoje Benko, Abigail Sellen, Richard Banks, Kenton O'Hara, Gavin Smyth, and William Buxton. 2016. Pre-Touch Sensing for Mobile Interaction. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 2289–2302. doi:10.1145/2858036.2858095

[19] Christian Holz and Patrick Baudisch. 2010. The Generalized Perceived Input Point Model and How to Double Touch Accuracy by Extracting Fingerprints. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)*. ACM, New York, NY, USA, 581–590. doi:10.1145/1753326.1753413

[20] Christian Holz and Patrick Baudisch. 2011. Understanding Touch. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 2501–2510. doi:10.1145/1978942.1979308

[21] Christian Holz and Patrick Baudisch. 2013. Fiberio: a touchscreen that senses fingerprints. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology* (St. Andrews, Scotland, United Kingdom) *(UIST '13)*. Association for Computing Machinery, New York, NY, USA, 41–50. doi:10.1145/2501988.2502021

[22] Christian Holz, Senaka Buthpitiya, and Marius Knaust. 2015. Bodyprint: Biometric User Identification on Mobile Devices Using the Capacitive Touchscreen to Scan Body Parts. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 3011–3014. doi:10.1145/2702123.2702518

[23] Christian Holz and Marius Knaust. 2015. Biometric Touch Sensing: Seamlessly Augmenting Each Touch with Continuous Authentication. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology* (Charlotte, NC, USA) *(UIST '15)*. Association for Computing Machinery, New York, NY, USA, 303–312. doi:10.1145/2807442.2807458

[24] Zeyuan Huang, Cangjun Gao, Haiyan Wang, Xiaoming Deng, Yu-Kun Lai, Cuixia Ma, Shengfeng Qin, Yong-Jin Liu, and Hongan Wang. 2024. SpeciFingers: Finger Identification and Error Correction on Capacitive Touchscreens. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 8, 1, Article 8 (2024), 28 pages. doi:10.1145/3643559

[25] Huy Viet Le, Thomas Kosch, Patrick Bader, Sven Mayer, and Niels Henze. 2018. PalmTouch: Using the Palm as an Additional Input Modality on Commodity Smartphones. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 360, 13 pages. doi:10.1145/3173574.3173934

[26] Huy Viet Le, Sven Mayer, and Niels Henze. 2018. InfiniTouch: Finger-Aware Interaction on Fully Touch Sensitive Smartphones. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology (UIST '18)*. ACM, New York, NY, USA, 779–792. doi:10.1145/3242587.3242605

[27] Huy Viet Le, Sven Mayer, and Niels Henze. 2019. Investigating the feasibility of finger identification on capacitive touchscreens using deep learning. In *Proceedings of the 24th International Conference on Intelligent User Interfaces* (Marina del Ray, California) *(IUI '19)*. Association for Computing Machinery, New York, NY, USA, 637–649. doi:10.1145/3301275.3302295

[28] Piyawat Lertvittayakumjorn, Shanqing Cai, Billy Dou, Cedric Ho, and Shumin Zhai. 2024. Can Capacitive Touch Images Enhance Mobile Keyboard Decoding?. In *The 37th Annual ACM Symposium on User Interface Software and Technology (UIST '24)*. ACM, New York, NY, USA, 17 pages. doi:10.1145/3654777.3676420

[29] Chentao Li, Jingze Yu, Ke He, Jianjiang Feng, and Jie Zhou. 2024. SwivelTouch: Boosting Touchscreen Input with 3D Finger Rotation Gesture. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 8, 2 (2024), 30 pages. doi:10.1145/3659584

[30] Zhuojun Li, Chun Yu, Yizheng Gu, and Yuanchun Shi. 2023. ResType: Invisible and Adaptive Tablet Keyboard Leveraging Resting Fingers. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23)*. ACM, New York, NY, USA, 14 pages. doi:10.1145/3544548.3581055

[31] Xinshuang Liu, Yizhong Zhang, and Xin Tong. 2024. V-Hands: Touchscreen-based Hand Tracking for Remote Whiteboard Interaction. In *The 37th Annual ACM Symposium on User Interface Software and Technology (UIST '24)*. ACM, New York, NY, USA, 1–14. doi:10.1145/3654777.3676412

[32] Damien Masson, Alix Goguey, Sylvain Malacria, and Géry Casiez. 2017. Leveraging Finger Identification to Integrate Multi-Touch Command Selection and Parameter Manipulation. *International Journal of Human-Computer Studies* 99 (2017), 36 pages. doi:10.1016/j.ijhcs.2016.12.008

[33] Fabrice Matulic, Daniel Vogel, and Raimund Dachselt. 2017. Hand Contact Shape Recognition for Posture-Based Tabletop Widgets and Interaction. In *Proceedings of the 2017 ACM International Conference on Interactive Surfaces and Spaces (ISS '17)*. ACM, New York, NY, USA, 3–11. doi:10.1145/3132272.3134126

[34] Sven Mayer, Huy Viet Le, and Niels Henze. 2017. Estimating the Finger Orientation on Capacitive Touchscreens Using Convolutional Neural Networks. In *Proceedings of the 2017 ACM International Conference on Interactive Surfaces and Spaces (ISS '17)*. ACM, New York, NY, USA, 220–229. doi:10.1145/3132272.3134130

[35] Sven Mayer, Xiangyu Xu, and Chris Harrison. 2021. Super-Resolution Capacitive Touchscreens. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)*. ACM, New York, NY, USA, 1–10. doi:10.1145/3411764.3445703

[36] Tobias Meyer and Dominik Schmidt. 2010. IdWristbands: IR-based user identification on multi-touch surfaces. In *ACM International Conference on Interactive Tabletops and Surfaces* (Saarbrücken, Germany) *(ITS '10)*. ACM, New York, NY, USA, 277–278. doi:10.1145/1936652.1936714

[37] Gianpaolo Palma, Narges Pourjafarian, Jürgen Steimle, and Paolo Cignoni. 2024. Capacitive Touch Sensing on General 3D Surfaces. *ACM Trans. Graph.* 43, 4, Article 103 (July 2024), 20 pages. doi:10.1145/3658185

[38] Aditya Singh Rathore, Weijin Zhu, Afee Daiyan, Chenhan Xu, Kun Wang, Feng Lin, Kui Ren, and Wenyao Xu. 2020. SonicPrint: a generally adoptable and secure fingerprint biometrics in smart devices. In *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services* (Toronto, Ontario, Canada) *(MobiSys '20)*. ACM, New York, NY, USA, 121–134. doi:10.1145/3386901.3388939

[39] Jun Rekimoto. 2002. SmartSkin: an infrastructure for freehand manipulation on interactive surfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Minneapolis, Minnesota, USA) *(CHI '02)*. ACM, New York, NY, USA, 113–120. doi:10.1145/503376.503397

[40] Stephan Richter, Christian Holz, and Patrick Baudisch. 2012. Bootstrapper: recognizing tabletop users by their shoes. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Austin, Texas, USA) *(CHI '12)*. Association for Computing Machinery, New York, NY, USA, 1249–1252. doi:10.1145/2207676.2208577

[41] Simon Rogers, John Williamson, Craig Stewart, and Roderick Murray-Smith. 2011. AnglePose: Robust, Precise Capacitive Touch Tracking via 3D Orientation Estimation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 2575–2584. doi:10.1145/1978942.1979318

[42] Julia Schwarz, Robert Xiao, Jennifer Mankoff, Scott E. Hudson, and Chris Harrison. 2014. Probabilistic Palm Rejection Using Spatiotemporal Touch Features and Iterative Classification. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 2009–2012. doi:10.1145/2556288.2557056

[43] Robin Schweigert, Jan Leusmann, Simon Hagenmayer, Maximilian Weiß, Huy Viet Le, Sven Mayer, and Andreas Bulling. 2019. KnuckleTouch: Enabling Knuckle Gestures on Capacitive Touchscreens using Deep Learning. In *Proceedings of Mensch Und Computer 2019* (Hamburg, Germany) *(MuC '19)*. ACM, New York, NY, USA, 387–397. doi:10.1145/3340764.3340767

[44] Weinan Shi, Chun Yu, Xin Yi, Zhen Li, and Yuanchun Shi. 2018. TOAST: Ten-Finger Eyes-Free Typing on Touchable Surfaces. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 1 (2018), 23 pages. doi:10.1145/3191765

[45] Paul Streli and Christian Holz. 2021. CapContact: Super-Resolution Contact Areas from Capacitive Touchscreens. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)*. ACM, New York, NY, USA, Article 289, 14 pages. doi:10.1145/3411764.3445621

[46] Feng Wang, Xiang Cao, Xiangshi Ren, and Pourang Irani. 2009. Detecting and Leveraging Finger Orientation for Interaction with Direct-Touch Surfaces. In *Proceedings of the 22nd Annual ACM Symposium on User Interface Software and Technology (UIST '09)*. ACM, New York, NY, USA, 23–32. doi:10.1145/1622176.1622182

[47] Te-Yen Wu, Lu Tan, Yuji Zhang, Teddy Seyed, and Xing-Dong Yang. 2020. Capacitivo: Contact-Based Object Recognition on Interactive Fabrics using Capacitive Sensing. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology (UIST '20)*. ACM, New York, NY, USA, 13 pages. doi:10.1145/3379337.3415829

[48] Robert Xiao, Julia Schwarz, and Chris Harrison. 2015. Estimating 3D Finger Angle on Commodity Touchscreens. In *Proceedings of the 2015 International Conference on Interactive Tabletops and Surfaces (ITS '15)*. ACM, New York, NY, USA, 47–50. doi:10.1145/2817721.2817737

[49] Xiangyu Xu, Jiadi Yu, Yingying chen, Qin Hua, Yanmin Zhu, Yi-Chao Chen, and Minglu Li. 2020. TouchPass: towards behavior-irrelevant on-touch user authentication on smartphones leveraging vibrations. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking* (London, United Kingdom) *(MobiCom '20)*. ACM, New York, NY, USA, Article 24, 13 pages. doi:10.1145/3372224.3380901

[50] Xin Yi, Chen Wang, Xiaojun Bi, and Yuanchun Shi. 2020. PalmBoard: Leveraging Implicit Touch Pressure in Statistical Decoding for Indirect Text Entry. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. ACM, New York, NY, USA, 13 pages. doi:10.1145/3313831.3376441

[51] Yang Zhang, Michel Pahud, Christian Holz, Haijun Xia, Gierad Laput, Michael McGuffin, Xiao Tu, Andrew Mittereder, Fei Su, William Buxton, and Ken Hinckley. 2019. Sensing Posture-Aware Pen+Touch Interaction on Tablets. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. ACM, New York, NY, USA, 1–14. doi:10.1145/3290605.3300285

Table 10: Knuckle Touch accuracy, precision and recall on KnuckleTouch [43]. †Whether training split was included in pretraining.

Protocol	Setting	Accuracy	Precision	Recall
P1	CapFM frozen	0.92	**0.90**	0.90
Multi-task	KnuckleTouch [43]	**0.95**	0.89	**0.96**
P2	10% labels	0.91	0.90	0.86
Efficiency	25% labels	0.91	0.91	0.86
	50% labels	0.91	0.91	0.85
P2	Random init	0.72	0.36	0.50
Ablation	⌞ + finetune	0.87	0.84	0.85
	Mean pooling	0.91	0.89	0.87
	Plain MAE	0.92	0.91	0.91
P3	KnuckleTouch	**0.95**	**0.95**	0.92
KnuckleTouch	CapFM			
protocol	⌞ KnuckleTouch data excluded†	**0.95**	0.94	**0.93**
	⌞ training split included†	0.93	0.93	0.90

Appendix

A Comparing model complexities

Table 11 shows a comparison of model complexity across relevant architectures for capacitive image processing. Computation is measured in multiply–accumulate operations (MACs) and corresponding floating-point operations (FLOPs = 2× MACs), omitting BatchNorm and activation overhead due to its negligible cost.

B Implementation Details

We summarize the full set of hyperparameters governing the model architecture, contact auxiliary loss, and training procedure in Table 12. We also provide the equations used to derive the pseudo-contact masks.

B.1 Quantile thresholding and soft saliency

Let q be the τ-quantile of the pixel values in x:

$$q \;=\; \mathrm{Quantile}(\mathrm{vec}(x),\, \tau)\,.$$

A soft saliency map is then computed by applying a sigmoid to the thresholded and temperature-scaled frame:

$$s \;=\; \sigma\Big(\frac{x-q}{T}\Big), \qquad \sigma(z) = \frac{1}{1+e^{-z}}.$$

This yields $s \in (0,1)^{H\times W}$.

B.2 Gradient magnitude term (Sobel filters)

A gradient magnitude map is computed using Sobel filters (as in the code). Define the Sobel kernels:

$$K_x = \frac{1}{8}\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \qquad K_y = \frac{1}{8}\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}.$$

Using 2D convolution with padding=1 (so the output remains $H \times W$):

$$g_x = x * K_x, \qquad g_y = x * K_y,$$

and the gradient magnitude:

$$g = \sqrt{g_x^2 + g_y^2 + \varepsilon}, \qquad \varepsilon = 10^{-8}.$$

Finally, the gradient map is normalized by its spatial maximum:

$$\tilde{g} = \frac{g}{\max_{u,v} g(u,v) + \delta}, \qquad \delta = 10^{-6},$$

so that $\tilde{g} \in [0,1]^{H\times W}$.

C Additional Evaluation on KnuckleTouch

We further evaluated CapFM on the KnuckleTouch dataset [43]. KnuckleTouch is a finger vs. knuckle classification task. We evaluate the model using the same three protocols. Table 10 reports accuracy, precision, and recall for all evaluation settings.

Under P1, the frozen pretrained encoder slightly outperforms the original KnuckleTouch model (95% vs. 92%) while maintaining comparable precision and recall. Performance remains stable with only 10% of the downstream labels. Random initialization performs substantially worse, while the plain MAE baseline is competitive but generally falls short of the full model. Under KnuckleTouch's original evaluation protocol, CapFM remains competitive with and without KnuckleTouch included during pretraining, indicating that the learned representation generalizes beyond the pretraining datasets.

Table 11: Comparison of neural network architectures for capacitive image processing. FLOPs = 2×MACs. BatchNorm and activation FLOPs omitted.

Model	Input	Architecture	Params	MACs	FLOPs	Source
TSI	16×18	Multinomial logistic regression	∼8K	negligible	negligible	arXiv
Orientation	15×22	3-layer CNN + Dense	2.6M	23.2M	46.4M	Analytical (code†)
CapFingerID	15×27	4-layer CNN + BN + Dense	853K	35.9M	71.8M	Analytical (code†)
TouchPose	41×72	Multi-task U-Net encoder-decoder (3 heads)	4.94M	472M	944M	`eth-siplab/TouchPose`
CapContact (generator)	72×41	WGAN-GP: VGG-style FCN generator	861K	11.33G	22.66G	`eth-siplab/CapContact`
CapContact (gen.+critic)	72×41	WGAN-GP: VGG generator + VGG critic	5.24M	5.78G	11.56G	`eth-siplab/CapContact`
SpeciFingers	3×224×224 RGB ×49 frames	AlexNet (frozen) + FC + 2-layer LSTM	17.45M (14.98M train.)	32.85G	65.7G	`iscas-MMSketch/SpeciFingers`
SpeciFingers (inference)	50 × 3 × 90 × 120	4-layer CNN + LSTM	5.1M	1.90G	3.81G	`iscas-MMSketch/SpeciFingers`
V-Hands (first frame)	2×96×128 cap. image + mask	U-Net + SE + 6-scale GRU + IK solver	29.16M	6.50G	13.00G	`V-Hands/V-Hands`
V-Hands (with GRU)	2×96×128 cap. image + mask	U-Net + SE + 6-scale GRU + IK solver	29.16M	∼13.73G	∼27.46G	`V-Hands/V-Hands`
CapFM MAE (training)	1×64×64	ViT-S MAE (p=8): encoder+decoder + pseudo-contact aux head	12.07M	0.960G	1.919G	-
CapFM MAE (encoder inference)	1×64×64	ViT-S encoder only (p=8), no decoder/contact head	10.70M	0.692G	1.384G	-

† Parameter count extracted from source code (GitHub); not reported in the original paper.

Model Architecture	
Model type	`mae_contact`
Backbone	ViT-Small (`vit-small`)
Input size	64×64, $C = 1$
Patch size	8 ($8 \times 8 = 64$ patches)
Encoder depth	6
Embedding dim	384
Num heads	8
MLP ratio	4
Mask ratio	0.75
Decoder depth	4
Decoder embed dim	128
Decoder num heads	4

Contact Auxiliary	
`contact_loss`	`bce_dice`
`focal_alpha`	0.25
`focal_gamma`	2
`dice_weight`	0.45
`lambda_contact`	0.15
`contact_warmup_epochs`	10
`pseudo_quantile`	0.82
`pseudo_saliency_temp`	0.04
`pseudo_gradient_weight`	0.45

Training & Optimization	
Batch size	64
Epochs	150
Optimizer	AdamW
Learning rate	3×10^{-4}
Weight decay	0.05
Grad clip norm	1
Mixed precision	`true`
Workers	4
Sampling seed	42
Dataset sampling	`balanced`
LR schedule	cosine
Warmup epochs	5
Min LR ratio	0.01

Table 12: MAE-contact hyperparameters side by side: model architecture, contact auxiliary, and training/optimization.