Universiteit Leiden

Probing, Attribution, and Attention in a Siamese Mouse-Dynamics Biometric Encoder

Wesley Gahr (s2589710)

Supervisor:
Dr. Christos Athanasiadis

Bachelor Thesis– Data Science & AI

Leiden Institute of Advanced Computer Science (LIACS)
liacs.leidenuniv.nl

July 1, 2026

Abstract

Behavioural biometrics verify a person by how they behave, such as how they move a mouse, rather than by a password. We study one such model, a Siamese mouse-dynamics verifier (SMDV): an encoder that turns a mouse-movement session into a 128-number embedding and verifies a user by how close a new session sits to that user's earlier sessions. Two questions are asked: what the model relies on, and how far we can trust the answer. A single interpretability tool can give a confident but wrong answer, so we pair every result with a control that tests whether it is real. We use three lenses. Probing classifiers measure what each layer makes readable. Integrated Gradients measure the input events that drive the verification decision. We test attention by erasing the events it points at and watching the decision. Each finding is checked against untrained networks of the same shape, a codelength measure, a gradient-free occlusion measure, and deletion tests. We find that identity is the main learned signal, and that the final embedding also genuinely encodes a family of motor-kinematic traits such as jerk and reversals, while timing traits such as pausiness are not. Attention lines up with the decision but does not drive it. Ranking input features by attribution size reflects their scale, not the model. Attribution rises towards the end of a session mostly because of the recurrent architecture, and the learned effect pulls it earlier. The main lesson is methodological: for this kind of model, faithfulness and the model's understanding must be measured with controls, not assumed.

Contents

1 Introduction **1**
- 1.1 The situation . . . 1
- 1.2 Problem statement . . . 1
- 1.3 Research questions . . . 1
- 1.4 Contributions . . . 2
- 1.5 Thesis overview . . . 2

2 Background **3**
- 2.1 The model under study . . . 3
- 2.2 Behavioural biometrics from mouse dynamics . . . 3
- 2.3 Interpretability primitives . . . 5

3 Related Work **5**
- 3.1 Probing and compositional attribution . . . 5
- 3.2 Attention as explanation . . . 5
- 3.3 Faithfulness, normalization, and time-series attribution . . . 6
- 3.4 The gap . . . 6

4 Approach **6**
- 4.1 Probing methodology . . . 6
- 4.2 Attribution for the verification decision . . . 9
- 4.3 IG-on-probe: localising where traits are read from . . . 12
- 4.4 Attention analysis: correlation and causal erasure . . . 12
- 4.5 How we validate . . . 13

5 Experiments **13**
- 5.1 Data and evaluation setup . . . 13
- 5.2 RQ1: What is represented, and where? . . . 14
- 5.3 RQ2: What drives verification, and is it faithful? . . . 15
- 5.4 RQ3: Where are traits read from? . . . 19
- 5.5 RQ4: Is attention an explanation? . . . 19

6 Conclusions and Further Research **21**
- 6.1 Synthesis . . . 21
- 6.2 Methodological lessons . . . 21
- 6.3 Limitations . . . 21
- 6.4 Future work . . . 22

A Behavioural trait definitions **22**

B Per-trait probing results **24**

References **27**

1 Introduction

1.1 The situation

Banks and other online services need to know who is using an account. One way to keep checking, without asking the user for anything, is behavioural biometrics: recognising a person by how they behave rather than by a password. Mouse dynamics is one such signal. The way a person moves the cursor, where they pause, and how they click is fairly stable for that person and hard for someone else to copy [19]. This makes it useful for spotting deviating behaviour, such as when a user is getting frauded.

There is a reason these patterns are personal. Hand movement follows regular motor rules: it tends to be smooth [13] and it trades speed against accuracy in a predictable way [12]. The timing of how a person moves their mouse differs from person to person [24]. A model can learn these patterns and use them to tell users apart.

The model studied in this thesis is one such verifier, built at ThreatFabric for anti-fraud in online banking. It turns a short sequence of mouse events into an embedding with 128 dimensions, and decides whether a new session belongs to a user by how close it sits to that user's earlier sessions.

1.2 Problem statement

The verifier works, but it is hard to see why. It is a neural network that compresses a session into 128 numbers, trained only to place same-user sessions close together and different-user sessions far apart. Nothing in it states, in terms a person can read, what it pays attention to. For a system that helps decide whether to trust a banking session, that opacity is a problem. Concretely, we want to say which events in a session, and which input features, push the verification distance up or down, and to be sure that answer reflects the model rather than an artefact of how we measured it.

The obvious approach is to apply a standard interpretability tool and read off the answer. That can mislead. Attention weights look like they show what a model uses, but don't always reflect what affects the model's output [18]. Gradient-based importance on a recurrent network leans towards the end of the sequence whatever the model learned [17]. A single tool, taken at face value, can give a confident explanation that is wrong.

This thesis therefore asks two questions together: what the verifier relies on, and how far we can trust the answer. Every result is paired with a check that tests whether it is real or an artefact of the method.

1.3 Research questions

We study one central question: *what does a Siamese mouse-dynamics verifier rely on, and how do we know the answer is trustworthy?* It breaks into:

- **RQ1 (representation).** What behavioural information is linearly decodable from the encoder's layers, and where in the network?
- **RQ2 (decision).** Which input events and features drive the verification distance, and is that attribution faithful, not a method artefact?

- **RQ3 (attribution).** *Where in the input* are specific behavioural traits read from?
- **RQ4 (attention).** Does the model's attention faithfully indicate what it uses, or is it only correlated with it?

1.4 Contributions

1. **A faithfulness-first study.** To our knowledge this is the first interpretability study of a Siamese behavioural-biometric *verifier*, whose decision is a squared distance to a stored profile rather than a class label. The one rigorous attribution method we know of for Siamese encoders [26], an attribution method designed for dot-product similarity between two encoded inputs, does not fit this squared-distance, centroid-gallery setup (Section 3). Every result is paired with a control that tests whether it is real, not an artefact of the method.

2. **Findings about the model.** Identity is the main signal the encoder learns, and it clearly beats untrained networks of the same shape (randomly initialised encoders of the same architecture, the baseline that shows what training *added* rather than what the wiring passes through on its own). The final embedding also genuinely encodes a family of motor-kinematic traits, such as jerk, reversals and speed, beyond what an untrained network already preserves. Timing traits such as pausiness are not reliably kept. The verification decision draws on the whole session, leaning a little towards the later events, though that lean is mostly structural rather than learned.

3. **Methodological cautions, each shown on the SMDV rather than assumed.** Attention here is correlated with attribution but does not drive the decision, so it has to be tested by intervention [18]. Ranking features by attribution size reflects each feature's scale, not the model: an attribution inherits the units of the feature it scores, so such a ranking just follows scale [23]. The rise of attribution towards the end of a session is mostly a structural property of the recurrent readout [17]. The part that is learned runs the other way, pulling attribution earlier. Probe accuracy has to be judged against a random-weight network and a codelength measure, not against chance.

4. **A compositional attribution method (IG-on-probe).** We introduce IG-on-probe, which composes a trained trait probe with Integrated Gradients to attribute a trait back to the input mouse events, in the spirit of Concept Gradient [5]. Held to the same faithfulness standard as the rest of the work, its cross-feature attribution proves not to be weight-specific (an untrained encoder reproduces it), so we present it as a method with that honest negative rather than a claim about where traits are read from.

1.5 Thesis overview

This work was carried out as a bachelor thesis at LIACS in collaboration with ThreatFabric, supervised by Dr. Christos Athanasiadis. Section 2 gives the necessary background. Section 3 reviews related interpretability work and the gap. Section 4 describes our methods. Section 5 reports the experiments as finding/validation pairs. Section 6 concludes.

Feature	Meaning	σ (dataset)
`event_type`	event kind (move / click / scroll)	0.20
`event_seq_change_flag`	event-type change flag	0.37
`angle`	direction of movement	0.70
`y_displacement`	vertical movement	111
`x_displacement`	horizontal movement	154
`total_displacement`	movement magnitude	162
`time_diff`	time since the previous event (ms)	3.6×10^5

Table 1: The seven input features recorded per mouse event, ordered by their dataset-wide standard deviation σ. The σ values span about seven orders of magnitude. This is why a cross-feature attribution ranking follows feature scale rather than what the model learned, and is reported only as a scale effect (RQ2).

2 Background

2.1 The model under study

This thesis studies one trained model, a *Siamese mouse-dynamics verifier* (SMDV). Its job is to check whether the person using the mouse is the real account owner, from the way they move it. The model reads a session of up to 100 mouse events. Each event has seven numbers (Table 1): the event type, whether that type just changed, the x, y and total distance moved, the time since the last event, and the movement angle. From these the model produces a list of 128 numbers, called an embedding. This embedding is a fingerprint of how the person moves. Inside, the events pass through a masking step, normalisation, two attention layers and two LSTM layers (Figure 1). Each attention layer has several *heads*: parallel attention sub-computations that can each focus on different positions. A residual *skip connection* runs around each attention block, adding its input back onto its output so information can bypass it. The encoder is *Siamese*: the same encoder, with the same weights, is applied to each session, and two sessions are compared by the distance between their embeddings. It was trained with a metric-learning objective, a triplet loss that pulls two sessions from the same person together in embedding space and pushes sessions from different people apart [31]. The effect is that two sessions from the same person get similar embeddings, and sessions from different people get different ones. To check a new session, the model measures how far its embedding is from the user's stored average of enrolled sessions, the *gallery centroid* $\bar{g}$. This (squared Euclidean) distance is written d^2. A small distance means the same person.

2.2 Behavioural biometrics from mouse dynamics

Mouse dynamics means telling people apart by *how* they move and click the mouse. Surveys of the area find the useful signals fall into a few groups: timing, the direction of movement, and clicking [19]. The seven features above fit these groups. These signals are not random. Moving a mouse is a motor skill, and people do it in a stable, personal way. Smooth movement follows the minimum-jerk pattern [13], and aimed movement has a fast part followed by small corrections, as described by Fitts' law [12]. How a person differs from these patterns stays much the same from one session to

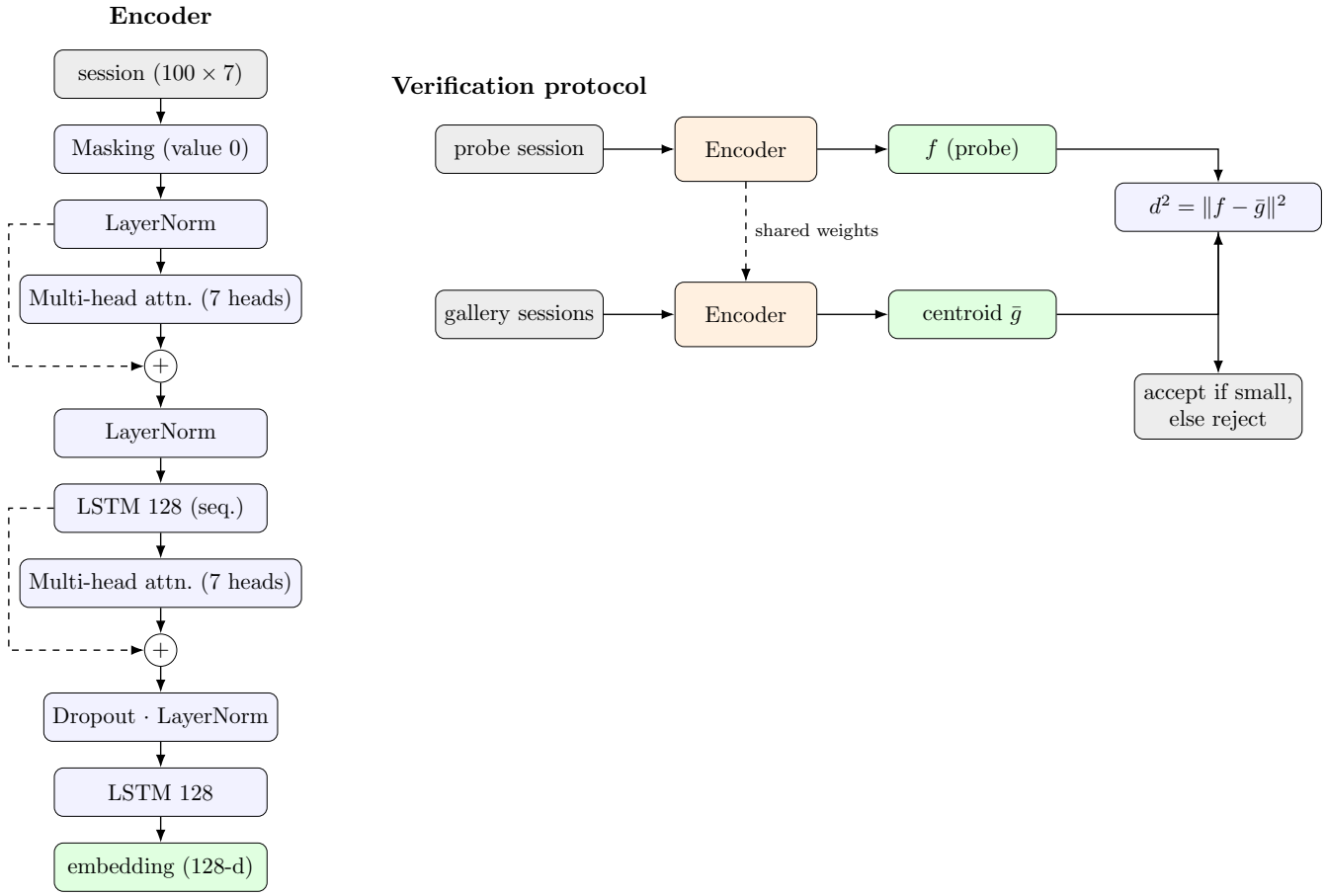


Figure 1: The Siamese mouse-dynamics encoder (left) and verification protocol (right). The encoder maps a session of up to 100 mouse events, each with seven features, to a 128-dimensional embedding; the two multi-head attention blocks each use a residual skip connection (dashed) around them, and `Masking` ignores zero-padded positions. At verification, a probe session and the user's stored gallery sessions pass through the *same* (shared-weight) encoder; the squared Euclidean distance d^2 between the probe embedding f and the gallery centroid $\bar{g}$ is thresholded to accept or reject.

the next. Mouse movement is also linked to attention and personality [24], so traits like hesitation and speed are likely to carry identity rather than just noise.

2.3 Interpretability primitives

Three tools are used in this work. A *probe* is a small, simple classifier trained on the model's frozen inner numbers for a session (its *representation* of that session: the 128-number embedding, or an earlier layer's activations), to see whether some property can be read off them [6]. *Integrated Gradients* scores how much each input value pushed the model's output up or down. By design these scores add up to the change in the output, a property called Completeness [33]. *Attention weights* show where each layer looks. To test whether they really matter, we delete the events a method calls important and see if the output changes [27]. We also check whether an explanation still holds when the model's weights are randomised [2]. This last check is about *faithfulness*: an explanation is faithful if it reflects what the model actually computes rather than merely looking plausible, so a faithful explanation should change when the model does.

Two further tools recur in the validation. *Occlusion* replaces an event or feature with a neutral baseline and measures the change in the output, a gradient-free counterpart to the deletion test above. Probe accuracy also has a ceiling: once two representations both let the probe score near-perfectly, accuracy can no longer tell which one encodes a property more cleanly. An *MDL* (minimum-description-length) *codelength* is a finer ruler that still can: it counts, in bits, how much information it takes to pin down the labels once you already hold the representation. A representation that exposes the property more cleanly makes the labels easier to predict, so fewer bits are needed, and the measure keeps discriminating after accuracy has maxed out [35].

3 Related Work

3.1 Probing and compositional attribution

Probes are a common way to study what a network has stored [6]. But a probe can pick up on patterns that are not really there, so its score must be compared against a control in which the labels are shuffled [14]. Running probes layer by layer, to see where information shows up, is a known approach [34]. The closest work to ours joins a probe with gradient attribution to ask *where* in the input a property is read from, shown for text in BERT [25]. This sits in a line of concept-direction methods: TCAV represents a concept as a direction in a layer's activations and tests how sensitive the decision is to moving along that direction [20], and Concept Gradient combines such a direction with the model's gradient to attribute the decision to the concept [5]. We use Integrated Gradients instead of plain gradients, and attribute the probe's output to the input rather than the decision to a concept.

3.2 Attention as explanation

It is debated whether attention weights explain a decision. Some work shows that attention often does not match what the model actually uses, and that a different attention pattern can give the same answer [18]. Other work argues that, tested carefully, attention can be a fair explanation [36]. A middle approach removes the high-attention events and checks the effect, treating the size of an

attention weight as a rough sign of importance [32]. Attention rollout combines attention across layers into one map from input to output [1]. We do not take a side in this debate in advance. Instead we test, on the SMDV, whether attention reflects what the decision uses.

3.3 Faithfulness, normalization, and time-series attribution

A separate question is whether an attribution can be trusted at all. One check randomises the model's weights: a faithful explanation should change when the model itself changes, so if the explanation barely moves after randomisation, it was never really explaining the model [2]. Other checks measure how well an attribution predicts the model's real change in output when features are removed [4, 37], including remove-and-retrain tests such as ROAR [15] and its retraining-free successor ROAD [29]. Comparing features against each other is harder still. Because an attribution inherits the units of the feature it scores, features recorded on very different scales cannot be ranked against each other by raw attribution size [22, 23]. A different kind of attribution sidesteps this by measuring each feature's contribution to the model's output rather than scoring the input [8]. Finally, gradient methods, including the Integrated Gradients we use, fade over time on LSTMs and over-weight late events [17], and time-series saliency is unreliable enough to need validation [16], so we also use a gradient-free check.

3.4 The gap

Two gaps remain. First, this kind of interpretability has rarely been used on a *Siamese* behavioural-biometric model: most work on biometric explainability scores features for a classifier, not for a verifier whose decision is a distance to a stored profile. The one rigorous attribution method we know of for Siamese encoders [26] is built for dot-product similarity between two encoded inputs, and does not transfer to our setting, where the decision is a squared distance to a centroid averaged over several stored sessions rather than to a single encoded input. Second, the warnings above (attention may not faithfully reflect the decision, rankings of features by attribution size can follow their scale rather than the model, and gradient methods on recurrent models over-weight late events) are usually studied one at a time, not together. This thesis does both: it studies a real mouse-dynamics verifier with probing, attribution and attention, and pairs every result with the check that tests it.

4 Approach

4.1 Probing methodology

Fourteen behavioural traits taken from the literature [19, 24] were chosen to test which ones can be read off each layer of the encoder, and where. During probing the encoder was kept frozen and only the probe was trained on its activations, so the probe measures what the model has already put into those activations, not a new representation of its own [6].

Each trait is a single number computed from a session's raw event sequence before any labelling, as a simple summary of the seven input features over the real (non-padding) events. For example, `speed` is the mean per-event displacement divided by the time since the previous event; `reversals` is the fraction of consecutive events whose movement angle turns by more than 90°; `jerk` is the

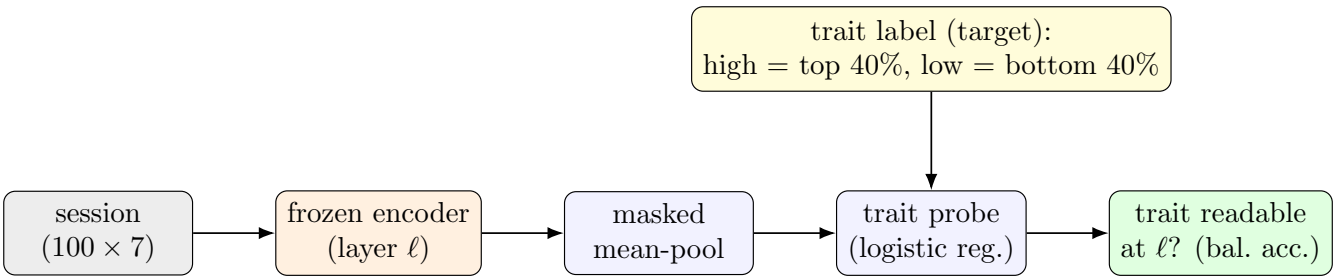
standard deviation of the movement angle; and `stroke_count` sums the event-type-change flag. Appendix A defines all fourteen.

Each trait was turned into a simple two-way label. A session counts as high if its trait value is above the 60th percentile and low if it is below the 40th. The middle 20% is dropped to avoid borderline cases. The two groups are equal in size, so chance is exactly 0.5. A few traits cannot be split this way (`event_density`, for example, is capped at 100 events, so its 40th and 60th percentiles are equal), and these were left out.

One probe was trained for each trait at each layer. For the layers that give one value per event, those values were averaged over the real (non-padding) events into a single vector per session. This *masked mean-pool* is the same input the probe is later explained on. Probes were trained at five points: the raw input (`raw_input_mean`), after each attention block (`ln_after_attn1`, `ln_after_attn2`) and at each LSTM output (`lstm_12` and `lstm_13`, the last being the final embedding). Each probe is a logistic-regression classifier, scored by balanced accuracy so that an uneven split cannot inflate it [7]. The probing procedure is summarised in Figure 2.

To stop the probe from cheating, the data were split so that no user appears in both training and test (`GroupKFold` with $k = 5$ on `user_id`). The score then reflects the trait, not a remembered person. Identity probing is different: it needs a user's other sessions in training, so it uses session-level folds instead, which is noted wherever the two are compared.

One more check guards against a probe that scores well just because it can fit anything. The same probe was retrained on labels shuffled *within* each user, and selectivity is the gap between the real-label and shuffled-label accuracy, selectivity $= \text{acc}_{\text{real}} - \text{acc}_{\text{control}}$ [14]. If a trait's shuffled-label accuracy is almost as high as its real one, the result is a quirk of the probe, not encoded information.



GroupKFold by user (no user in both train and test), repeated at each layer;
validated against a random-weight encoder, an MDL codelength, and a within-user shuffle (Section 4.5).

Figure 2: The probing process. At each layer ℓ, a session is passed through the frozen encoder, its activations are mean-pooled over the real events, and a logistic-regression probe is trained to predict a binary trait label (high vs. low: the top and bottom 40% of the trait's values). The probe's balanced accuracy measures whether the trait is linearly readable at that layer. Folds are split by user, and every reading is validated against the controls of Section 4.5.

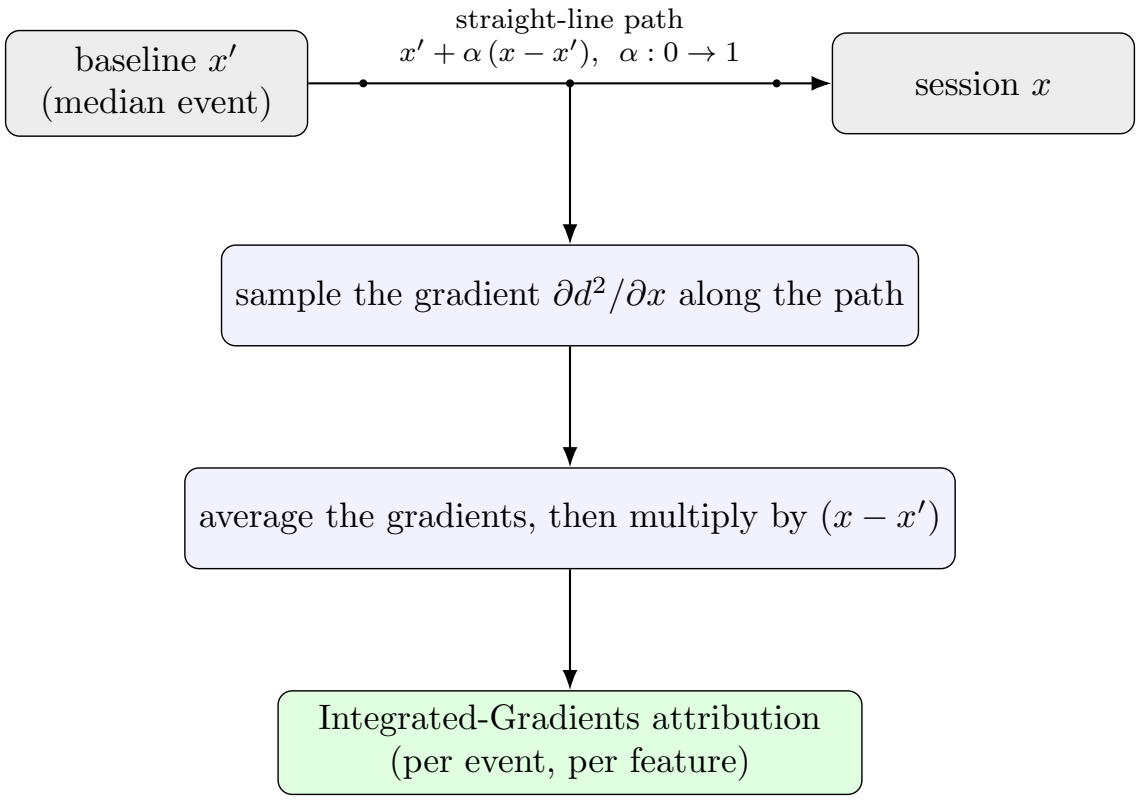
4.2 Attribution for the verification decision

The decision under study is the verification distance from Section 2: the squared distance d^2 between a session's embedding and the user's stored average, the gallery centroid $\bar{g}$. A small distance is accepted, a large one rejected. We study this squared Euclidean distance because it is the metric the encoder was trained under: the triplet loss operates on squared Euclidean distances, so attributing d^2 targets the same geometry the model learned. We want to know which events, and which features, push this distance up or down.

To find out, Integrated Gradients were computed for d^2 [33], the quantity the deployed verifier actually thresholds, so the attribution explains the decision the system makes rather than the training objective. The method gives each input value a score by adding up the model's gradient along a straight line from a baseline to the real session. Its Completeness property means the scores add up exactly to the change in d^2 from the baseline, so their sizes can be compared on the output scale.

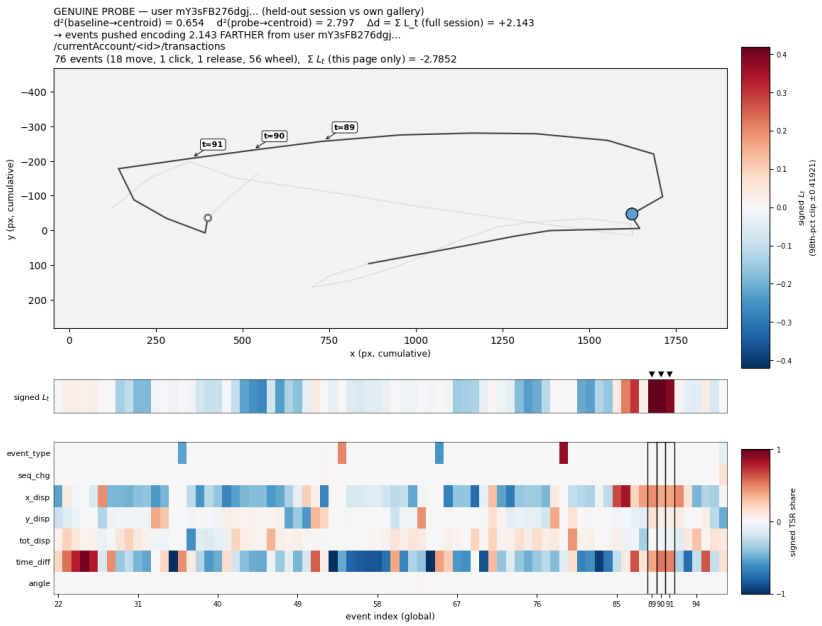
Choosing the baseline needs care because the encoder's `Masking` layer treats an all-zero event as padding. A naive all-zero baseline would therefore mark every position as padding, break the integration path, and void Completeness. Instead, the baseline preserves each session's real and padding structure: real positions are set to the dataset-median value of each feature (a neutral, "average" event), while padding positions stay zero. The number of integration steps was chosen so that the Completeness gap stays below 5%. Figure 3 summarises the method.

The raw scores were used directly as the attribution of each feature at each timestep. Figure 4 shows one genuine and one impostor session three ways. The top panel is the cursor path reconstructed from the session, with each event coloured by how much it moves the verification distance. The middle strip is the same per-event score in sequence. The bottom heatmap splits each event's score across the seven input features, so it shows which feature carries the attribution and when.

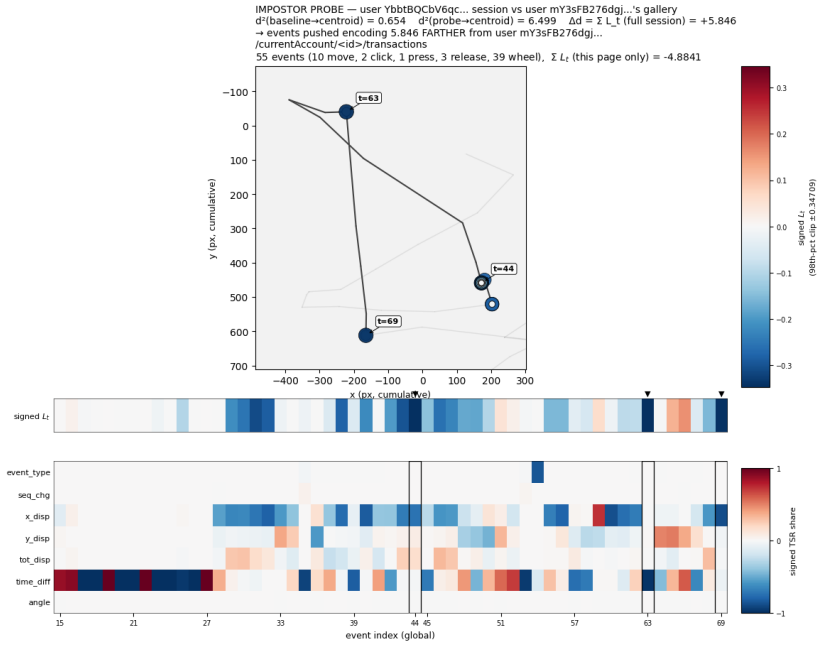


Completeness: the attributions sum to $d^2(x) - d^2(x')$.

Figure 3: How Integrated Gradients attributes the verification distance. The gradient of d^2 with respect to the input is sampled along the straight-line path from the per-query baseline x' to the session x, averaged, and multiplied by $(x - x')$. By the Completeness property the resulting per-event, per-feature attributions sum exactly to $d^2(x) - d^2(x')$.



(a) Genuine session: the probe sits close to the user's own centroid (small distance).



(b) Impostor session: the probe is far from the claimed user's centroid (large distance). The panels read the same way as in (a).

Figure 4: Integrated-Gradients attribution for a genuine (a) and an impostor (b) session, each shown three ways: the reconstructed cursor path coloured by each event's signed contribution to the verification distance (top), the same per-event contributions in sequence (middle), and each event's contribution split across the seven input features (bottom).

4.3 IG-on-probe: localising where traits are read from

Probing shows whether a trait is readable at a layer, but not where in the input it is read from. To find the where, the trait probe was composed with the encoder and attributed with Integrated Gradients (Figure 5). Read forward, a session passes through the frozen encoder up to the probed layer, the activations are mean-pooled over the real events, and the trait probe turns that pooled vector into a trait logit. Integrated Gradients then attributes this logit back to the $(100, 7)$ input, along a straight-line path from the same per-query baseline used for the verification attribution above, giving a per-event, per-feature map of where the trait is read from. Because the gradient runs through the encoder and the probe together, this is a concept-direction attribution in the sense of Section 3: it follows Concept Gradient [5], which attributes the model's own decision to a concept, except that here we attribute only the probe's output, as in earlier probe-with-gradient work [25].

The per-feature map this produces is not reported as a result. Under the random-weight check of Section 4.5, an untrained encoder gives almost the same map, so it reflects the architecture and the trait's definition, not anything the model learned. The method is kept as a working pipeline that respects Completeness, but the question of *where* a trait is read from is left open.

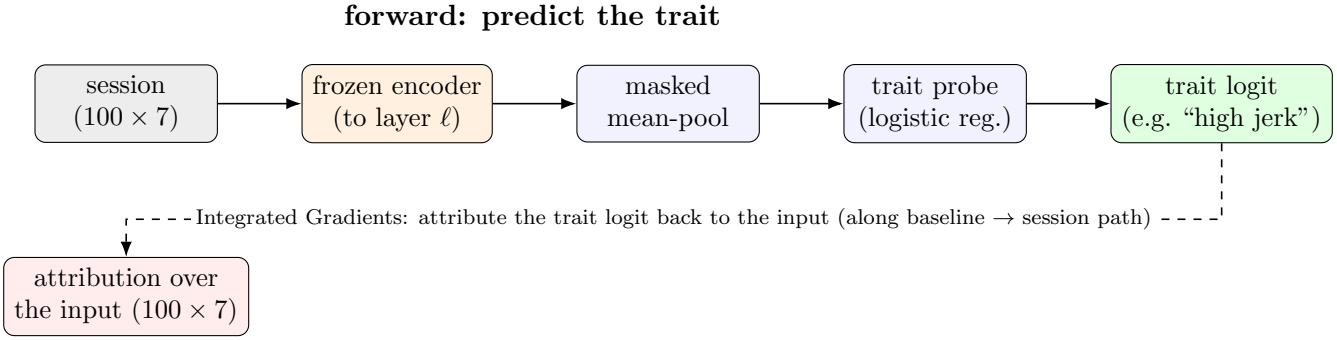


Figure 5: The IG-on-probe pipeline. Forward (top): a session passes through the frozen encoder to the probed layer, is mean-pooled over its real events, and a logistic-regression trait probe produces a trait logit. Integrated Gradients (dashed) then attributes that logit back to the (100×7) input, along a straight-line path from a per-query baseline, giving a per-event, per-feature map of where the trait is read from.

4.4 Attention analysis: correlation and causal erasure

The two attention layers (seven heads each) produce weights that are often read as where the model looks. Whether these weights reflect what the decision actually uses is debated [18, 36, 32], so we test it directly rather than assume it. For each real event position, we took the average attention it receives from the other events, and compared this against the per-event Integrated-Gradients score $\sum_f |\mathrm{IG}(t, f)|$ from the previous section, where $\mathrm{IG}(t, f)$ is the attribution of feature f at event t, so the sum is one event's total importance. Agreement was measured with Spearman's rank correlation [2] over real events only, since the padding would otherwise flood the correlation with identical zeros.

A correlation between attention and attribution still does not show that attention is *what the decision uses*, so a causal test was added. The erasure idea [32] ranks events by how important

a method says they are, removes the top ones, and measures how much the decision changes. Following it as a deletion curve [30, 27], the events a ranking calls important were replaced with the baseline session and the rise in d^2 was measured (the deletion area under the curve). We report the gain over a random ranking on the same sessions, for Integrated Gradients and for each attention ranking, so a method gets credit only if deleting its top events hurts the decision more than deleting random ones.

4.5 How we validate

After each measurement above we ask one question: what could produce this same result *without* the effect being real? We then run the control that rules that alternative out. Only findings that survive their control are reported. Four such checks recur across the three lenses.

The first guards against a probe that fits noise rather than a real trait: the selectivity control above, the gap between real-label and within-user-shuffled accuracy [14].

To tell what training added from what the architecture passes through on its own, each probe was also run on five untrained encoders of the same shape [2], and the trained score was compared against that spread with a paired bootstrap interval [10]: the test sessions are resampled with replacement many times and the trained-minus-untrained gap is recomputed on each resample, giving a confidence range for that gap (*paired* because both the trained and the untrained encoders are scored on the same resampled sessions). Because accuracy maxes out on the easiest traits, we also report the minimum-description-length codelength, which keeps telling the encoders apart after accuracy no longer can [35].

A third check looks at the attributions themselves. To confirm an attribution reflects the model and not the method, we removed subsets of events and checked how well the attribution had predicted the real change in d^2 [4, 37]. We did not rank features against each other by raw size: an attribution carries each feature's units, so such a ranking is not meaningful [22, 23].

The last check separates a trait that is merely present from one the decision uses: the important events were removed and the effect on d^2 measured, by single-feature occlusion and by a deletion curve that erases the highest-ranked events first [27]. Gradient methods on recurrent models fade over time [17], so the gradient-free occlusion also guards against late-event bias. Removing a trait from the embedding to see whether the decision needs it (amnesic probing, [11]) is left to future work (Section 6).

This habit is why several results in the next chapter changed once they were checked. Two cautions apply throughout: a trait being readable does not mean the decision uses it [6], and conclusions from mean-pooled layers hold only under that pooling [3].

5 Experiments

5.1 Data and evaluation setup

All results were obtained on the Erste web-banking dataset: 15,000 sessions from 1,332 users, each session a sequence of mouse events recorded through the same SDK as the training data. The model was trained on data from a different *frontend*: a different web application the tracking SDK was embedded in, with its own page layout and interaction flow. It is evaluated here on Erste, so

every result is an out-of-distribution test: the seven input features (Table 1) are the same, but the web interface that produced them differs. This is deliberate. A verifier in production routinely meets frontends it was not trained on, so an interpretation of the model is only useful if it holds under that shift, which is exactly what evaluating on a different frontend checks.

Before interpreting the encoder, we checked that it transfers to this new frontend at all. In the 128-dimensional embedding, sessions from the same user are closer together than sessions from different users: the mean intra-user distance is 1.81 and the mean inter-user distance is 2.06, giving a ratio of 0.88. The encoder therefore still separates users on a frontend it was not trained on, which is what makes the interpretation that follows meaningful.

5.2 RQ1: What is represented, and where?

We first asked what the encoder holds, and at which layer. For identity, a probe (a small classifier trained on a layer's frozen activations) was trained at each layer to predict which of 313 users a session belongs to (chance about 0.3%). A higher probe accuracy means the layer makes identity more readable. Identity is easiest to read after the first LSTM layer. It then dips through the second attention block, is brought back by the skip connection, and is squeezed again at the final embedding (Table 2). So the information is there all along, but the final layer keeps only what verification needs.

To tell what the encoder learned from what its wiring passes through on its own, identity probing was repeated on five untrained encoders of the same shape (Section 4.5, Table 2). Identity survives this check: at the final embedding the trained encoder reads identity at about 2.2 times the untrained rate, and its score (0.071) lies far outside the spread of all five untrained encoders (0.029–0.035). It is more than double even the best of them. So identity is a genuinely learned signal, which is what the training was meant to produce.

	Trained		Untrained (5 seeds)		
Layer	acc.	×ch.	acc.	×ch.	Ratio
raw input (mean)	0.082	26	0.082	26	1.00
after LSTM-1	0.205	64	0.138 ± 0.002	43	1.48
after Attention-2	0.199	62	0.145 ± 0.003	45	1.37
final embedding	0.071	22	0.032 ± 0.002	10	2.24

Table 2: Identity-probe accuracy by layer (313 users, chance $\approx 0.3\%$): the trained encoder versus five random-initialised encoders of the same architecture (mean $\pm$ s.d. across seeds). Identity peaks after the first LSTM and is compressed at the final embedding, yet the trained encoder beats every one of the five untrained encoders at all learned layers. The raw-input row is identical by construction (a pipeline sanity check).

The behavioural traits give a more mixed picture (Figures 6 and 7; the full per-trait numbers are in Table 5, Appendix B). The test throughout is to compare the trained encoder against an untrained one of the same shape: only a gap between them shows that *training* made a trait readable, rather than the architecture merely passing the input through. At the middle layers there is no such gap: an untrained encoder reads the traits about as well as the trained one, so readability

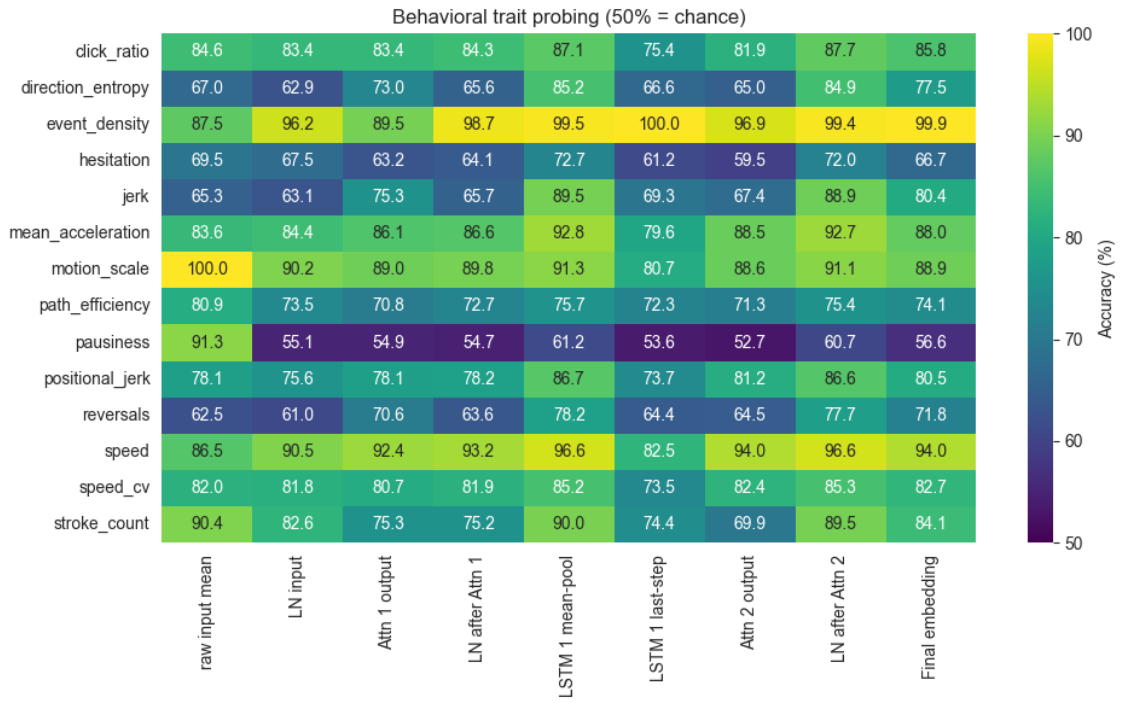


Figure 6: Behavioural-trait probing accuracy across layers (chance = 50%).

there reflects preserved input, not anything learned. The final embedding is different. There the trained encoder reads a family of motor-kinematic traits markedly better than the untrained ones. The gap for jerk is +0.112 (90% interval [+0.101, +0.122], about fifteen times the spread across untrained encoders), with direction entropy, reversals, speed, positional jerk, stroke count and mean acceleration also clearly above the untrained baseline (Table 5). The same traits also need roughly 300 to 750 fewer bits to describe (the codelength measure of Section 4.5, averaged over orderings) and pass the selectivity check, so all three checks agree. The timing and intention traits behave differently: pausiness and hesitation show only a small accuracy gap that the codelength does not back up (an untrained encoder describes their labels in fewer bits, not more), and path efficiency shows no gap at all, so none of them is reliably encoded.

So the encoder learns a specific family of motor-kinematic traits at its final layer (the kind the motor-control literature describes, from minimum-jerk movement to Fitts-style corrections [13, 12]), on top of its main identity signal. This is about what can be *read* from the embedding, not what the decision *uses*: a readable trait is not necessarily what the verification step relies on.

5.3 RQ2: What drives verification, and is it faithful?

We next asked which events and features drive the verification distance, and whether the attribution can be trusted. Integrated Gradients were computed for d^2 across many sessions. The per-feature sizes (Figure 8) come in the same fixed order across sessions, but that order just follows the features' raw scales, not anything the model learned, so we report it as a scale effect, not as which feature the model uses.

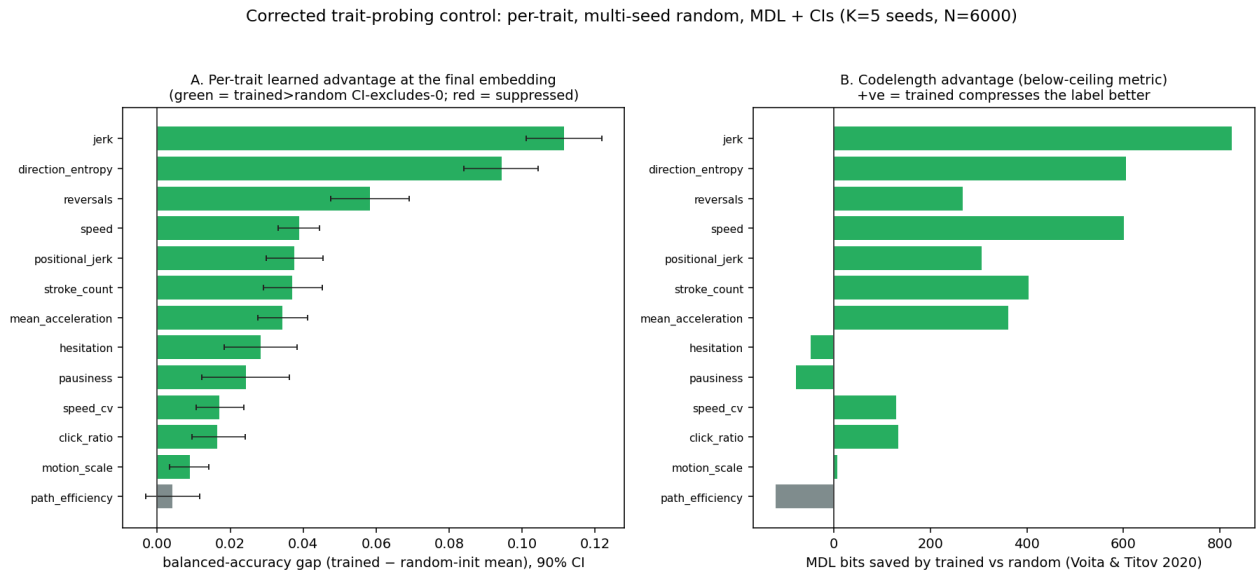


Figure 7: Corrected per-trait control at the final embedding. Left: balanced-accuracy gap between the trained encoder and a five-seed random-initialisation distribution, with 90% bootstrap intervals (green = genuinely learned, red = not reliably encoded). Right: bits saved by the trained encoder under the minimum-description-length measure.

Three checks pin down what the attribution can and cannot say. The first is faithfulness: do the per-event scores really predict the model's behaviour? Each Integrated-Gradients score is one event's contribution to d^2, so summing the scores over a set of events gives the change the attribution predicts if those events were removed. For many random subsets we compared this predicted change against the real change in d^2 when the same events are replaced by the baseline: the two track each other closely (correlation about 0.83 across subsets), as Completeness implies, so the per-event scores are faithful [4, 37]. The second check concerns ranking the seven features against one another by attribution size. Under the random-weight control this ranking has the same top feature for the trained and an untrained encoder [2], so it simply follows the features' scales, not anything the model learned. This is why such a ranking is not meaningful [22, 23], and we make no claim about which single feature matters most. The third check is timing (Figures 9 and 10). Attribution rises towards the second half of a session, and a gradient-free occlusion profile agrees with the Integrated-Gradients one (rank correlation about 0.70 on well-covered positions), so the rise is not just the known late-event bias of recurrent gradients [17]. Looking per feature refines this (Table 3): across five random-init seeds, the untrained encoders are *each* more late-weighted than the trained one (the trained *centre of mass* of attribution, its attribution-weighted average event position on a 0 = start to 1 = end axis, lies below all five for every feature), so the late-weighting comes mostly from the architecture. The part that is learned goes the other way: training pulls attribution a little *earlier* for every feature.

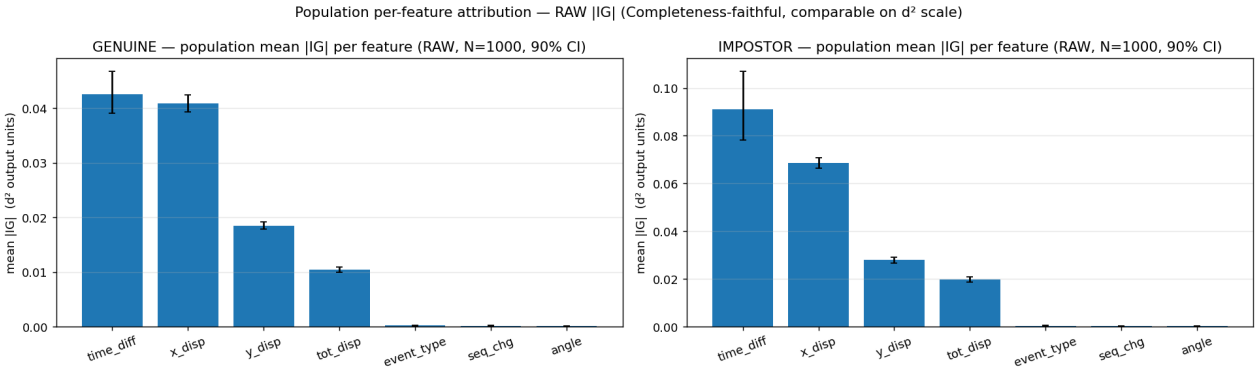


Figure 8: Population per-feature attribution (raw $|IG|$, $N = 1000$, 90% CI).

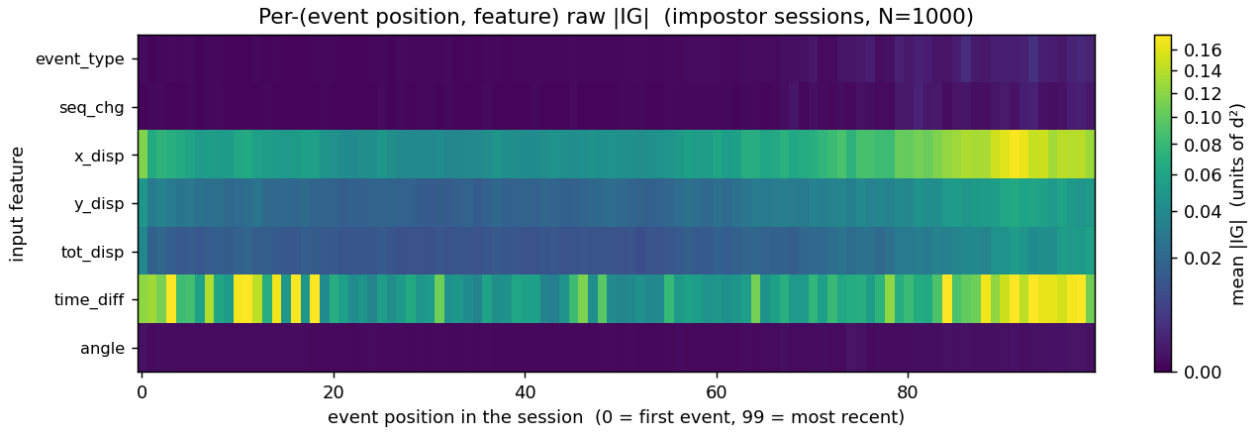


Figure 9: Per-(event position, feature) raw |IG| for impostor sessions ($N = 1000$). Magnitudes are in units of d^2 and comparable across sessions on the output scale (Completeness, [33]), but each feature carries its own units, so colour is *not* comparable across feature rows: the near-dark rows for the small-scale features (`event_type`, `event_seq_change_flag`, `angle`) reflect feature scale, not unimportance. The cross-feature order is a scale effect that survives weight randomisation [2] and is not a claim about which feature the model uses.

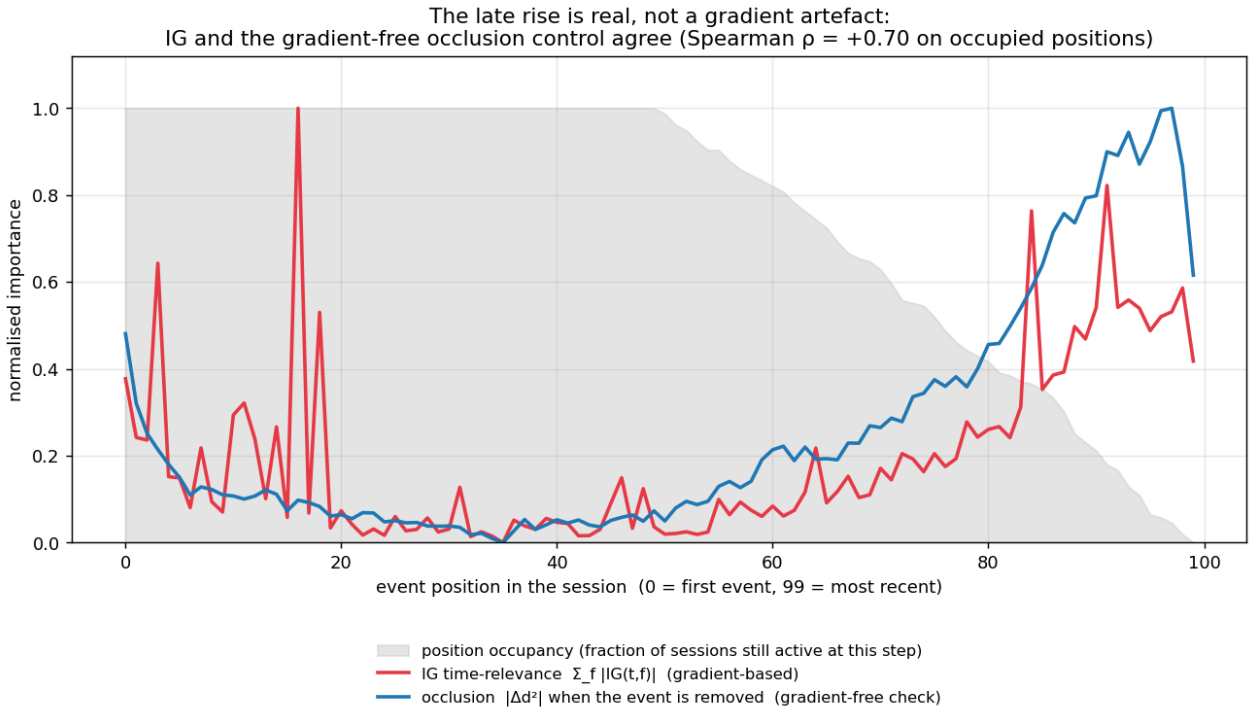


Figure 10: Recency-bias control (impostor sessions). The IG time-relevance profile $\sum_f |\mathrm{IG}(t, f)|$ (red, gradient-based) is compared against a gradient-free occlusion profile (blue, the change in d^2 when each event is replaced by the baseline); position occupancy (grey) is the fraction of sessions still active at each step. Both profiles are min–max scaled only for overlay. They agree on reliably-occupied positions (Spearman $\rho = +0.70$), so the rise of attribution towards the end of a session is a real temporal effect, not the gradient-vanishing artefact of recurrent models [17].

Feature	COM trained	COM untrained (5 seeds)	ΔCOM [90% CI]	seeds later
`event_type`	0.571	0.710 ± 0.029	-0.139 $[-0.161, -0.118]$	5/5
`event_seq_change_flag`	0.585	0.662 ± 0.039	-0.077 $[-0.099, -0.056]$	5/5
`x_displacement`	0.602	0.801 ± 0.011	-0.199 $[-0.214, -0.183]$	5/5
`y_displacement`	0.568	0.750 ± 0.038	-0.182 $[-0.197, -0.167]$	5/5
`total_displacement`	0.576	0.829 ± 0.009	-0.253 $[-0.271, -0.235]$	5/5
`time_diff`	0.584	0.796 ± 0.013	-0.212 $[-0.230, -0.194]$	5/5
`angle`	0.553	0.740 ± 0.034	-0.187 $[-0.206, -0.168]$	5/5

Table 3: Per-feature temporal centre of mass (COM, 0 = start, 1 = end of the session) of the occlusion attribution, $N = 120$ impostor probes. The untrained encoders are more late-weighted than the trained one for every feature (the architecture's recency bias), while training pulls the COM *earlier* ($\Delta\mathrm{COM} < 0$, 90% bootstrap interval excludes 0, and the trained COM lies below all five seeds): the learned de-skewing effect. "Seeds later" counts how many of the five untrained encoders are more late-weighted than the trained one.

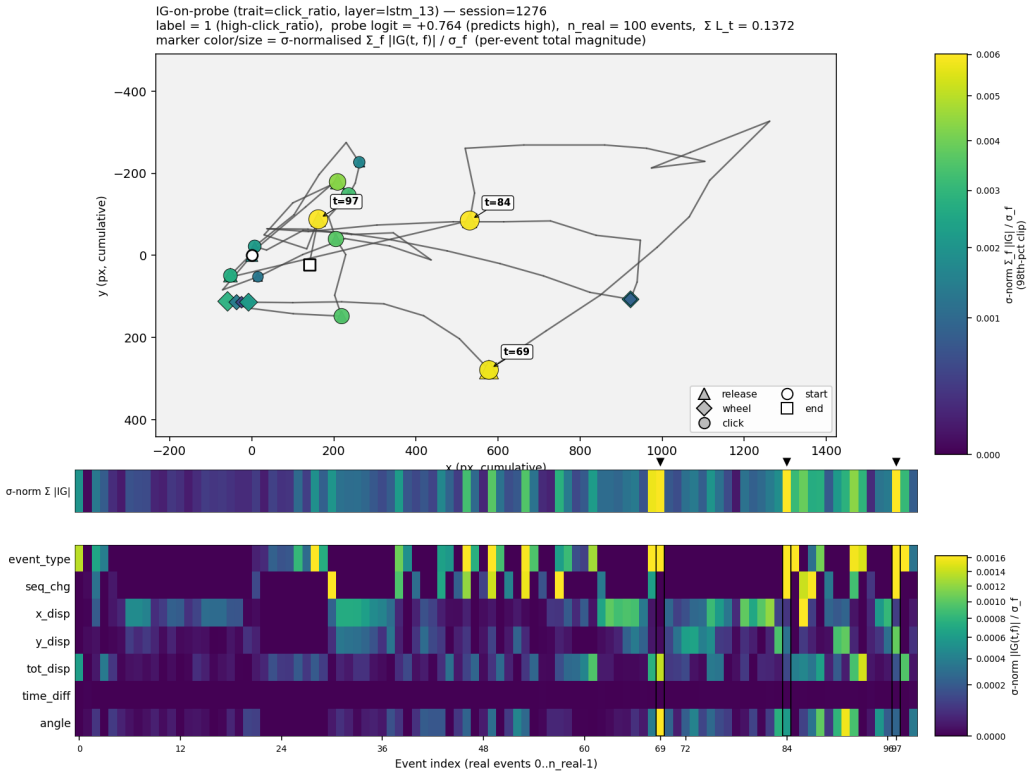


Figure 11: IG-on-probe for the click-ratio trait.

5.4 RQ3: Where are traits read from?

To ask *where* in a session a trait is read from, the trait probe was joined with Integrated Gradients, the IG-on-probe method of Section 4. On the click-ratio trait it works as intended (Figure 11): the probe is accurate, Completeness holds, and the feature the trait is built from comes out on top.

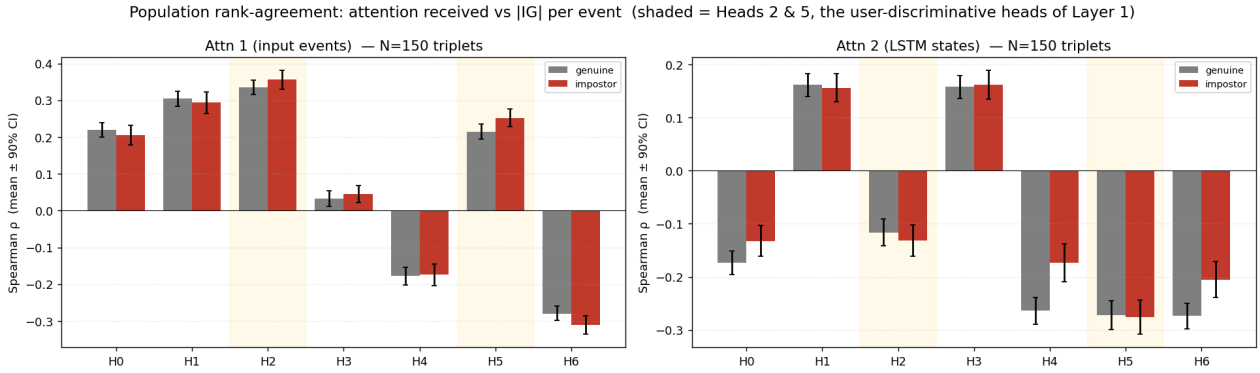
But the result does not survive the random-weight check. An untrained encoder gives almost the same per-feature map [2], so it reflects the architecture and how the trait is defined, not what the model learned. We therefore keep IG-on-probe as a working method that respects Completeness, but drop the claim about *where* a trait is read from. Testing this event by event is left for future work.

5.5 RQ4: Is attention an explanation?

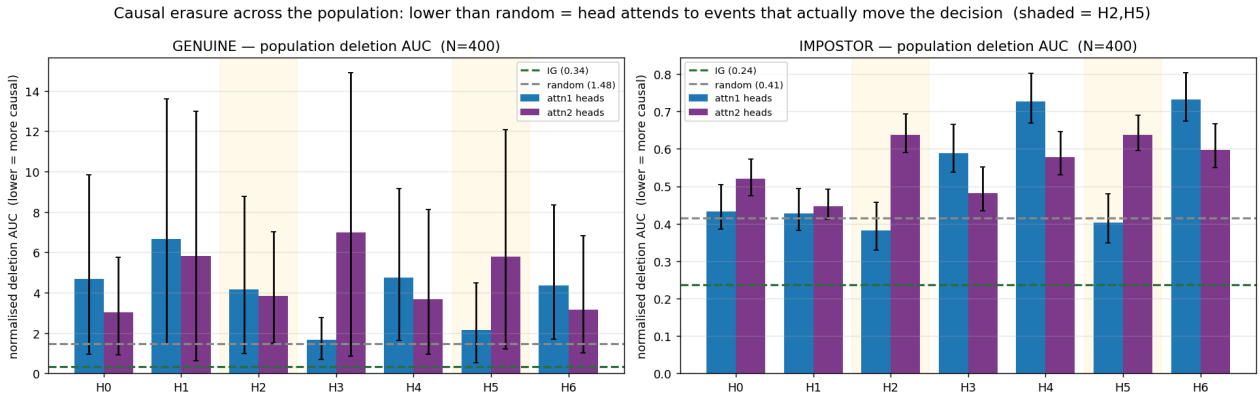
Finally we asked whether the model's attention shows what its decision uses. Across the data, the input-layer attention agrees moderately with the per-event Integrated-Gradients scores (rank correlation about +0.34), while the second attention layer, which works on mixed recurrent states, mildly disagrees with it (Figure 12a). The agreement depends on the head: the two heads that best separate users are also among the most aligned with the scores, while two others point the opposite way. The genuine and impostor curves sit on top of each other, so this is a fixed property of the model, not something that appears only for fakes.

Agreement is not use, so the rankings were tested by erasure (Figure 12b). The events a ranking

calls important were replaced with the baseline session, and the rise in d^2 was measured as a deletion curve [32, 27]. Removing the top Integrated-Gradients events hurts the decision far more than removing the top attention events: the gain over a random ranking is +0.178 for Integrated Gradients against +0.033 for the best single head (about five times as much), and erasing the second attention layer's choices is worse than random. So on this model attention goes along with the attribution but does not actually drive the decision. This verdict rests on the causal deletion test, the change in d^2 when events are removed, rather than on treating Integrated Gradients as ground truth, so it is evidence that the gradient signal is the more dependable per-event guide here, not an assumption that it is. This matches, on the SMDV, the cautionary findings of the attention-as-explanation debate [18, 32].



(a) Per-head rank agreement between attention and $|IG|$ (population).



(b) Causal erasure: deleting high-attention events vs high-IG events.

Figure 12: Attention versus Integrated Gradients: (a) per-head rank agreement, and (b) causal erasure, deleting high-attention versus high-IG events.

6 Conclusions and Further Research

6.1 Synthesis

This thesis asked what a Siamese mouse-dynamics verifier relies on, and how far the answer can be trusted.

First, what the model relies on. It is mainly a learned identity encoder. It still tells users apart on a frontend it was never trained on (RQ1), and identity is the one signal that clearly beats an untrained network. On top of that, the final embedding genuinely encodes a family of motor-kinematic traits (jerk, direction changes, reversals and the like) which an untrained network does not, while timing traits such as pausiness are not reliably kept. What drives a single decision (RQ2) is spread across the session and leans a little towards the later events, though most of that lean comes from the architecture, not from training. Attention (RQ4) lines up loosely with these signals but does not drive the decision.

Second, how far to trust it. Every result was paired with a check, and several changed once checked. The first reading, that the model encodes no behavioural traits, turned out to be an averaging artefact. The apparent source from IG-on-probe (RQ3) did not survive an untrained encoder and was dropped. And a ranking of features by attribution size turned out to follow each feature's scale, not the model. Results that survived are reported here, and the ones that did not are marked as withdrawn. The thesis therefore answers both sides of its question: what the verifier has learned, and how much of that we can trust.

6.2 Methodological lessons

A few lessons carry beyond the SMDV. Probe accuracy on its own says little: an untrained network of the same shape is already a strong feature extractor, so the baseline to beat is that network rather than chance. When accuracy hits its ceiling, a codelength measure still tells a learned representation apart from an untrained one. Attribution sizes cannot be ranked across features, because each carries its own scale. Compare them within a feature, or on the output scale instead. To ask which input matters, remove it and watch the decision. Attention should be tested by removing what it points at. Reading it off as an explanation is not enough. And on a recurrent model, any claim that importance grows over time has to be checked against a gradient-free measure, because the gradient itself fades. Running these checks is what turned three plausible findings into either a validated result or a withdrawn one.

6.3 Limitations

This study has clear limits. Only one trained model was studied, so the findings are about this encoder, not the architecture in general. Probing shows what is readable, not what the decision uses, so the trait results speak to what the embedding *contains*, not what verification *relies on*. The uncertainty estimates are also modest: the trait and temporal gaps come with bootstrap intervals, and the identity gap is reported as the trained score against the spread of five random initialisations, but all three rest on only five seeds, a coarse basis for the standard deviations behind the reported σ-multiples that more seeds would tighten. Several attribution results also depend on the choice of baseline, which is a modelling decision rather than a fixed quantity. Finally, the erasure test

is cleaner on impostor sessions than genuine ones, so the causal attention result is read from the impostor side.

6.4 Future work

Several directions follow naturally, in rough order of importance.

- *Test which traits the decision actually uses, not just carries.* Probing (Section 5) shows what is readable, not what is relied on. Amnesic probing [11, 28] would close this gap: it erases a trait's information from the embedding and measures how much verification degrades, so a trait the model truly uses should hurt the decision when it is removed. The hard part is doing this without being fooled, because a trait can look "used" merely because it varies a lot; a sound version needs baselines that correct for this (variance-matched and whitened-distance), repeated over several random initialisations, plus a second concept-based method as a cross-check, adapting TCAV [20] to the verifier.
- *Rank features by their effect on the output, not by raw attribution size.* Importances measured in the model's output, such as predictive-power importance [8] or distribution-based attribution [21], sidestep the scale problem discussed in RQ2.
- *Widen the attribution analysis.* The IG-on-probe pipeline could be run across all traits and layers rather than the subset shown here, and the temporal analysis could use a learned-perturbation method [9] in place of fixed occlusion.

A Behavioural trait definitions

Each trait is a single scalar computed from a session's real (non-padding) mouse events, using the seven input features of Table 1. Table 4 gives the definition used for each. `event_density` is listed for completeness but excluded from the probing results, because it is capped at the maximum session length and its 40th and 60th percentiles coincide (Section 4).

Trait	Definition (per session, over the real events)
`speed`	mean of `total_displacement` divided by `time_diff`
`mean_acceleration`	mean of $\lvert\Delta\,\text{speed}\rvert$ between consecutive events
`speed_cv`	std(speed)/mean(speed), the coefficient of variation
`jerk`	standard deviation of the movement angle
`positional_jerk`	standard deviation of Δ^2 speed, the second difference of speed
`reversals`	fraction of consecutive events whose movement angle turns by more than 90°
`direction_entropy`	Shannon entropy of the movement-angle histogram (8 bins over $[-\pi, \pi]$)
`motion_scale`	mean of `total_displacement`
`path_efficiency`	straight-line distance $\sqrt{(\sum \Delta x)^2 + (\sum \Delta y)^2}$ divided by total path length $\sum$ `total_displacement`
`hesitation`	95th percentile of `time_diff`
`pausiness`	standard deviation of `time_diff`
`click_ratio`	fraction of events whose `event_type` differs from the session's modal type
`stroke_count`	sum of the `event_seq_change_flag`
`event_density`	number of real events (excluded; see above)

Table 4: Definitions of the fourteen behavioural traits, as computed from each session's raw mouse-event sequence before the binary labelling of Section 4.

B Per-trait probing results

Table 5 gives the full per-trait results at the final embedding summarised by Figure 7: the balanced-accuracy gap over the random-initialisation baseline, the codelength saved, and selectivity, for all fourteen traits.

Trait	Acc. gap [90% CI]	MDL bits	Selec.
Genuinely learned: motor-kinematic family			
`jerk`	+0.112 [+0.101, +0.122]	+752	+0.14
`direction_entropy`	+0.094 [+0.084, +0.104]	+581	+0.14
`reversals`	+0.058 [+0.048, +0.069]	+300	+0.09
`speed`	+0.039 [+0.033, +0.045]	+569	+0.15
`positional_jerk`	+0.038 [+0.030, +0.045]	+324	+0.11
`stroke_count`	+0.037 [+0.029, +0.045]	+418	+0.17
`mean_acceleration`	+0.034 [+0.028, +0.041]	+407	+0.13
Weak / inconclusive			
`speed_cv`	+0.017 [+0.011, +0.024]	+137	+0.12
`click_ratio`	+0.016 [+0.009, +0.024]	+148	+0.16
`motion_scale`	+0.009 [+0.003, +0.014]	+24	+0.14
Not reliably encoded			
`hesitation`	+0.028 [+0.018, +0.038]	−49	+0.06
`pausiness`	+0.024 [+0.012, +0.036]	−104	+0.03
`path_efficiency`	+0.004 [−0.003, +0.012]	−131	+0.14

Table 5: Per-trait results at the final embedding ($N \approx 4800$, balanced 0.4/0.6-quantile split). *Acc. gap*: balanced-accuracy gap between the trained encoder and the five-seed random-initialisation mean, with 90% paired-bootstrap interval. *MDL bits*: codelength saved by the trained encoder over the untrained mean (Voita & Titov 2020), averaged over 30 example orderings (typical s.d. 30–45 bits). *Selec.*: accuracy above a within-user label-shuffle control (Hewitt & Liang 2019). A trait counts as genuinely learned only when all three agree: `pausiness` and `hesitation` have a small accuracy gap but *no* codelength advantage, and `path_efficiency` passes selectivity yet has neither a gap nor a codelength advantage, so none is reliably encoded. The degenerate `event_density` (tied quantiles) is excluded.

References

[1] S. Abnar and W. Zuidema. Quantifying attention flow in transformers. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020.

[2] J. Adebayo, J. Gilmer, M. Muelly, I. Goodfellow, M. Hardt, and B. Kim. Sanity checks for saliency maps. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. arXiv:1810.03292.

[3] Y. Adi, E. Kermany, Y. Belinkov, O. Lavi, and Y. Goldberg. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. In *International Conference on Learning Representations (ICLR)*, 2017. arXiv:1608.04207.

[4] M. Ancona, E. Ceolini, C. Öztireli, and M. Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. In *International Conference on Learning Representations (ICLR)*, 2018. arXiv:1711.06104.

[5] A. Bai, C.-K. Yeh, N. Y. C. Lin, P. Ravikumar, and C.-J. Hsieh. Concept gradient: Concept-based interpretation without linear assumption. In *International Conference on Learning Representations (ICLR)*, 2023. arXiv:2208.14966.

[6] Y. Belinkov and J. Glass. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics (TACL)*, 2019.

[7] K. H. Brodersen, C. S. Ong, K. E. Stephan, and J. M. Buhmann. The balanced accuracy and its posterior distribution. In *International Conference on Pattern Recognition (ICPR)*, 2010.

[8] I. Covert, S. Lundberg, and S.-I. Lee. Understanding global feature contributions with additive importance measures (SAGE). In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. arXiv:2004.00668.

[9] J. Crabbé and M. van der Schaar. Explaining time series predictions with dynamic masks. In *International Conference on Machine Learning (ICML)*, 2021. arXiv:2106.05303.

[10] B. Efron and R. J. Tibshirani. *An Introduction to the Bootstrap*. Chapman & Hall/CRC, 1993.

[11] Y. Elazar, S. Ravfogel, A. Jacovi, and Y. Goldberg. Amnesic probing: Behavioral explanation with amnesic counterfactuals. *Transactions of the Association for Computational Linguistics (TACL)*, 2021. arXiv:2006.00995.

[12] P. M. Fitts. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*, 47(6):381–391, 1954.

[13] T. Flash and N. Hogan. The coordination of arm movements: an experimentally confirmed mathematical model. *Journal of Neuroscience*, 5(7):1688–1703, 1985.

[14] J. Hewitt and P. Liang. Designing and interpreting probes with control tasks. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2019. arXiv:1909.03368.

[15] S. Hooker, D. Erhan, P.-J. Kindermans, and B. Kim. A benchmark for interpretability methods in deep neural networks (ROAR). In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[16] A. A. Ismail, M. Gunady, H. Corrada Bravo, and S. Feizi. Benchmarking deep learning interpretability in time series predictions. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. arXiv:2010.13924; introduces Temporal Saliency Rescaling (TSR).

[17] A. A. Ismail, M. Gunady, L. Pessoa, H. Corrada Bravo, and S. Feizi. Input-cell attention reduces vanishing saliency of recurrent neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. arXiv:1910.12370.

[18] S. Jain and B. C. Wallace. Attention is not explanation. In *NAACL-HLT*, 2019. arXiv:1902.10186.

[19] S. Khan, C. Devlen, M. Manno, and D. Hou. Mouse dynamics behavioral biometrics: A survey. *ACM Computing Surveys*, 56(6):1–33, 2024.

[20] B. Kim, M. Wattenberg, J. Gilmer, C. Cai, J. Wexler, F. Viegas, and R. Sayres. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In *International Conference on Machine Learning (ICML)*, 2018. arXiv:1711.11279.

[21] X. Li and K. M. Ting. Distribution-based feature attribution for explaining the predictions of any classifier. In *AAAI Conference on Artificial Intelligence (AAAI-26)*, 2026. arXiv:2511.09332.

[22] D. Lundstrom, T. Huang, and M. Razaviyayn. A rigorous study of integrated gradients method and extensions to internal neuron attributions. In *International Conference on Machine Learning (ICML)*, 2022. arXiv:2202.11912.

[23] D. Lundstrom and M. Razaviyayn. Four axiomatic characterizations of the integrated gradients attribution method. *arXiv preprint arXiv:2306.13753*, 2023.

[24] K. L. Meidenbauer, T. Niu, K. W. Choe, A. J. Stier, and M. G. Berman. Mouse movements reflect personality traits and task attentiveness in online experiments. *Journal of Personality*, 91(2):413–425, 2023.

[25] H. Mohebbi, A. Modarressi, and M. T. Pilehvar. Exploring the role of BERT token representations to explain sentence probing results. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2021. arXiv:2104.01477.

[26] L. Möller, D. Nikolaev, and S. Padó. An attribution method for siamese encoders. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2023. arXiv:2310.05703.

[27] V. Petsiuk, A. Das, and K. Saenko. RISE: Randomized input sampling for explanation of black-box models. In *British Machine Vision Conference (BMVC)*, 2018.

[28] S. Ravfogel, Y. Elazar, H. Gonen, M. Twiton, and Y. Goldberg. Null it out: Guarding protected attributes by iterative nullspace projection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020. arXiv:2004.07667.

[29] Y. Rong, T. Leemann, V. Borisov, G. Kasneci, and E. Kasneci. A consistent and efficient evaluation strategy for attribution methods (ROAD). In *International Conference on Machine Learning (ICML)*, 2022. arXiv:2202.00449.

[30] W. Samek, A. Binder, G. Montavon, S. Lapuschkin, and K.-R. Müller. Evaluating the visualization of what a deep neural network has learned. *IEEE Transactions on Neural Networks and Learning Systems*, 28(11):2660–2673, 2017.

[31] F. Schroff, D. Kalenichenko, and J. Philbin. FaceNet: A unified embedding for face recognition and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[32] S. Serrano and N. A. Smith. Is attention interpretable? In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019.

[33] M. Sundararajan, A. Taly, and Q. Yan. Axiomatic attribution for deep networks. In *International Conference on Machine Learning (ICML)*, 2017. arXiv:1703.01365.

[34] I. Tenney, D. Das, and E. Pavlick. BERT rediscovers the classical NLP pipeline. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019.

[35] E. Voita and I. Titov. Information-theoretic probing with minimum description length. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2020. arXiv:2003.12298.

[36] S. Wiegreffe and Y. Pinter. Attention is not not explanation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP-IJCNLP)*, 2019.

[37] C.-K. Yeh, C.-Y. Hsieh, A. S. Suggala, D. I. Inouye, and P. Ravikumar. On the (in)fidelity and sensitivity of explanations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. arXiv:1901.09392.