

Quantifying the Value of Link Prediction and Behavioural Trust in Infrastructure-Free Mesh Routing

Slok Regmi

Department of Software Engineering
Nepal College of Information Technology
Lalitpur, Nepal
slok.221642@ncit.edu.np

Bhusan Thapa

Department of Software Engineering
Nepal College of Information Technology
Lalitpur, Nepal
bhusan@ncit.edu.np

Abstract—Infrastructure-free mesh networks must route over links that appear, degrade and vanish faster than any control plane can advertise them. A common proposal is to let a graph neural network (GNN) predict which links are about to fail and route around them pre-emptively. We build that system end to end — a seeded mobility-and-radio simulator, a deterministic multi-metric baseline, a 5,089-parameter GraphSAGE link-survival predictor, and a behavioural trust graph — and measure what each component is worth, reporting the components that do not pay as carefully as the ones that do. We first show that the choice of delivery metric decides the answer: in a correctly buffered store-and-forward mesh, eventual delivery saturates near 0.95 for every strategy we test and ranks hop-count routing above loss-aware routing, while delivery within a 2s deadline ranks them the other way round, the two curves crossing at about 9s. Measured on the deadline an application actually experiences, three findings stand out. First, the deterministic multi-metric cost is a failure: tuned on disjoint seeds it does not beat the single classical ETX metric (0.736 against 0.746), and an ablation shows why — removing its energy, battery or latency terms *improves* on-time delivery by up to 1.5 percentage points each. Only convexity in packet loss earns its place, worth 3.7 points, and only behavioural trust earns more. Second, link-survival prediction does pay, and an oracle bounds how much: perfect foreknowledge of the next 5s is worth 2.6 points, of which our 19.9kB model captures 83% (+2.9%, $p = 0.030$, with route breaks cut 43%). At a 20s horizon that headroom persists but our predictor cannot reach it, gaining nothing while the oracle gains 3.1 points — a limitation of the model, not of the idea. Third, the behavioural trust graph is the component that pays outright, recovering 8.8%, 18.1% and 24.3% of on-time delivery lost to 10%, 20% and 30% malicious relays that each silently drop 80% of transit traffic (all $p < 10^{-4}$), at a 0.95–0.97 detection rate, a false-positive rate below 0.03, and no significant cost when no attacker is present ($p = 0.701$), and it still pays against adaptive attackers that evade detection. We also derive and confirm the closed-form ultra-low-bandwidth frontier imposed by the 158-byte header-plus-cryptography floor. All code, seeds and results are released.

Keywords—mesh networks, MANET routing, graph neural networks, link prediction, trust and reputation, delay-tolerant networking, reproducibility

I. INTRODUCTION

When cellular and Internet infrastructure is unavailable — a disaster zone, a protest, a remote valley, an outage —

nearby devices can still reach each other over short-range radio. Making that useful requires multi-hop routing over a graph whose edges are created and destroyed by node motion, fading and battery drain, on a control plane too slow to keep up.

The standard reactive and proactive protocols [1]–[3] respond to link failure after it happens. A recurring proposal in recent work is to make the network *predictive*: model the topology as a dynamic graph, learn which links are about to break, and reroute before they do [4]–[6]. The proposal is intuitively compelling. It is also, as far as we can tell, rarely accompanied by a measurement of how much of the end-to-end gain is attributable to prediction itself rather than to the metric it is bolted onto.

This paper builds the whole stack and measures each layer separately. We implement NEXUS: a layered infrastructure-free communication system in which cryptography provides confidentiality and authenticity, a dynamic graph layer provides topology intelligence, and the two never overlap — the graph component decides *where* a packet goes and never sees its plaintext. On top of a seeded random-waypoint and log-distance-fading simulator we evaluate five routing strategies, a GNN link-survival predictor with a matched oracle upper bound, and a watchdog-derived trust graph, over 20 seeds per configuration with paired significance tests throughout, and with every weight fitted on a disjoint seed set.

Our contributions are:

- 1) **Evidence that the delivery metric, not the routing strategy, decides the ranking.** With correct store-and-forward buffering, eventual delivery saturates near 0.95 and places hop-count routing above loss-aware routing; on-time delivery inverts the order. We treat the deadline as an experimental variable and show the crossover explicitly (Section VIII-A).
- 2) **A negative result on multi-metric routing.** The full multi-metric cost, with weights fitted on disjoint seeds, loses to plain ETX (−1.3%), and its energy, battery and latency terms are individually *harmful* — removing them

gains up to 1.5 points each. Only convexity in loss pays, worth 3.7 points (Sections VIII-A, VIII-I).

- 3) **An oracle-bounded evaluation of predictive rerouting.** A 5,089-parameter GNN gains +2.9% on-time delivery ($p = 0.030$) and cuts route breaks 43%; an oracle shows the total available gain is 2.6 points, so the model captures 83% of what any predictor could. At a 20s horizon the oracle still gains 3.1 points while our model gains nothing, locating the remaining headroom precisely (Section VIII-E).
- 4) **A behavioural trust graph that pays, with its cost stated.** It recovers 8.8%/18.1%/24.3% of on-time delivery at 10%/20%/30% attacker density (all $p < 10^{-4}$) at a false-positive rate under 0.03, and costs nothing measurable in a clean network ($p = 0.701$) (Section VIII-F).
- 5) **A closed-form ultra-low-bandwidth frontier** induced by the 158-byte fixed header-plus-cryptography cost, derived and confirmed against a capacity sweep in which eventual delivery reads 0.883 for a network delivering 12% of its traffic on time (Section VIII-H).

We state throughout which results are simulation-only, and Section IX lists what would have to change for these numbers to transfer to hardware.

II. RELATED WORK

Classical ad-hoc routing. AODV [1], DSR [3] and OLSR [2] span the reactive/proactive design space and remain the reference baselines [7]. The single most consequential refinement to their metrics was ETX [8], which prices a link by its expected number of transmissions rather than by hop count; RPL [9] carries the same idea into low-power lossy networks. Our results reinforce how much of the achievable gain this one idea already captures.

Graph learning for networks. GCN [10], GraphSAGE [11] and GAT [12] established the message-passing architectures we build on, and temporal graph networks [13] extend them to evolving edge sets. In the networking domain, RouteNet [4] models network performance with a GNN, Almasan et al. [5] pair GNNs with deep reinforcement learning for routing optimisation, and Geyer and Carle [6] learn distributed routing protocols directly. These target wired or SDN-managed topologies with a controller; the infrastructure-free case — no controller, stale views, node motion — is where we place the predictor, and we measure it against an oracle rather than only against a non-learned baseline.

Misbehaviour and trust. The watchdog/pathrater mechanism [14] and reputation systems such as CONFIDANT [15] established that observed forwarding behaviour can identify non-cooperative relays; Karlof and Wagner [16] catalogue selective-forwarding and sinkhole attacks, and Douceur [17] the identity-multiplication attack that cryptographic identity (not behaviour) must address. We reuse this machinery unmodified and report both its benefit and its false-positive cost in the same experiment.

Delay tolerance. When the graph is partitioned, storing and forwarding later is the only option [18], [19]. Our simulator

TABLE I
MEASURED COST OF THE SECURITY LAYER (MEAN OF 200 OPERATIONS, PYTHON CRYPTOGRAPHY ON A DESKTOP CPU).

Payload	Sealed	On wire	Expansion	Seal	Open
64 B	188 B	222 B	$3.47\times$	0.26 ms	0.32 ms
200 B	324 B	358 B	$1.79\times$	0.27 ms	0.38 ms
1024 B	1148 B	1182 B	$1.15\times$	0.44 ms	0.75 ms

Fixed overhead: 34 B binary header + 124 B cryptographic expansion.

implements store-and-forward with TTL and expiry, which is why congestion in our low-bandwidth experiments manifests as latency rather than loss.

Cryptography. We invent nothing: X25519 [20], [21], HKDF-SHA256 [22], ChaCha20-Poly1305 [23] and Ed25519 [24]. The separation of concerns is a design commitment of this work: the learned components never participate in providing confidentiality, integrity or authentication.

III. SYSTEM ARCHITECTURE

NEXUS is layered top-down as applications, a priority layer, end-to-end authenticated encryption, graph intelligence (prediction and trust), routing, and a transport abstraction over BLE or Wi-Fi Direct. Two properties of that ordering matter here: no upper layer names a specific radio, and no learned component sits on the security path.

A. Identity and end-to-end security

Each device generates an Ed25519 signing key and an X25519 agreement key on first run; its identity is the SHA-256 digest of the signing public key truncated to 8 bytes, never a human-chosen name. A message is sealed at the source with an ephemeral X25519 exchange, HKDF-SHA256 key derivation and ChaCha20-Poly1305, and signed with Ed25519; the routing header is bound in as associated data, so a relay cannot swap the destination without invalidating the tag. Relays forward an opaque blob. Table I gives the measured cost.

B. Packet format

The wire header is a 34-byte fixed structure: version, TTL, hop count, priority, an 8-byte packet identifier, 8-byte source and destination identifiers, a 4-byte timestamp and a 2-byte lifetime. Replay and duplicate suppression use the packet identifier with expiry-based pruning. We note this explicitly because our first implementation serialised the header as JSON at 365 bytes — $7.95\times$ expansion for a 64-byte message — which alone would have moved the low-bandwidth frontier of Section VIII-H by more than a factor of two. Header encoding is not an implementation detail at these scales.

IV. DYNAMIC GRAPH AND ROUTING COST

The network is a time-varying graph $G_t = (V_t, E_t)$. Nodes carry battery, reliability, trust, load and mobility; edges carry RSSI, packet loss, latency, bandwidth, stability, energy cost, transport type and link age. A protocol may read only these observables — never node positions.

A. The cost function

Path delivery probability is multiplicative in the per-edge success probabilities, so a per-edge cost must be *convex* in packet loss; a weighted sum that is linear in loss systematically under-prices weak links. NEXUS-D therefore prices an edge (a, b) by its expected transmissions under bounded ARQ with R retries,

$$\text{ETX}(e) = \frac{1}{(1 - p_e^{R+1})(1 - p_e)}, \quad (1)$$

and adds the terms a pure link-quality metric cannot see:

$$\begin{aligned} c(a, b) = & w_{\text{etx}} \text{ETX}(e) + w_\ell \ell_e + w_{\text{en}} \varepsilon_e \\ & + w_{\text{rel}}(1 - \min(\rho_b, \sigma_e)) + w_{\text{bat}}(1 - \beta_b) \\ & + w_{\text{tr}}(1 - \tau_b), \end{aligned} \quad (2)$$

with ℓ_e latency, ε_e energy cost, ρ_b node reliability, σ_e link stability, β_b battery and τ_b trust. Routes are minimum-cost paths under (2). This deterministic baseline is retained permanently: every learned component must be shown to beat it.

B. Predictive routing

Let $\hat{p}_e$ be the predicted probability that edge e survives until the next control-plane refresh. Maximising the probability that an entire path survives means maximising $\prod_e \hat{p}_e$, i.e. minimising $\sum_e -\log \hat{p}_e$, which composes additively with (2):

$$c_G(a, b) = c(a, b) - w_{\text{pred}} \log \hat{p}_e. \quad (3)$$

w_{pred} is fitted per control-plane refresh interval on scenario seeds disjoint from every evaluation seed; Section VII-D gives the procedure and the fitted values.

V. LINK-SURVIVAL GNN

A. Task and features

For every currently-up edge, predict whether it is still up one refresh interval later. Inputs are strictly the observables carried in link-state advertisements. Each node gets eight features (degree, battery, mean and best incident RSSI, mean and best incident loss, mean link age, trust); each edge gets nine (RSSI, loss, latency, log link age, RSSI trend over the last three observations, a trend-validity flag, bandwidth and both endpoint degrees). The RSSI trend is what makes a departing neighbour visible before it departs [25].

B. Architecture and cost

Two mean-aggregating GraphSAGE layers [11] of width 32 produce node embeddings z ; an edge head scores $[z_u + z_v \parallel |z_u - z_v| \parallel \mathbf{e}_{uv}]$, a form symmetric in (u, v) as an undirected link requires. The model has 5,089 parameters (19.9 kB in float32) and runs in 0.49 ms per snapshot for a 30-node, 56-edge graph — small enough to sit on a phone, and re-evaluated only once per control-plane refresh, so it adds no transmitted bytes at all.

The predictor is trained on its own traces, on an axis independent of the routing evaluation: 30 scenario seeds for training, 10 for validation and a further 10 held out for the

quality figures reported here, all three disjoint from each other and from the routing evaluation and weight-fitting seeds. Throughout this paper, “evaluation seeds” always means the seeds on which routing is measured, never these. Optimisation is Adam at 3×10^{-3} for 40 epochs with binary cross-entropy, selecting the epoch with the best validation loss.

C. Predictor quality

The upper block of Table III shows prediction quality against the majority-class rate. At the 5 s horizon the model is strong; at 20 s the task itself becomes hard, which matters for Section VIII-B, where longer refresh intervals are exactly the regime in which prediction would be most useful.

VI. BEHAVIOURAL TRUST GRAPH

Each relay is scored by whether packets handed to it are subsequently forwarded, observed in promiscuous mode [14]. Trust is an exponentially weighted average with asymmetric rates — a drop moves the score twice as fast as a success — floored at 0.05 so that no node is permanently blacklisted, and folded into (2) through w_{tr} . Crucially, the observer cannot distinguish a malicious drop from an exhausted link-layer retry, and our simulator preserves that ambiguity: an ARQ failure downstream of a relay is charged to that relay. The false-positive rates in Section VIII-F are therefore real, not an artefact of a friendly observation model.

We stress the boundary: trust and prediction shape *routing decisions*. Confidentiality, integrity and authentication come from the cryptographic layer alone. A learned component is never permitted to be the reason a message is secure.

VII. EXPERIMENTAL SETUP

A. Simulator

Nodes move by random waypoint [25] in a square area, arriving exactly at each waypoint before pausing — an implementation detail that matters, since letting nodes overshoot at high speed silently drains the area and makes the topology non-stationary. RSSI follows a log-distance path loss model [26] with exponent 3.0, 40 dB reference loss at 1 m and AR(1) log-normal shadowing ($\sigma = 2$ dB) initialised from its stationary distribution; links exist above a -92 dBm sensitivity floor. Packet error rate is exponential in link margin, latency grows as the link weakens and with neighbour count, and capacity follows a rate-adaptation ladder up to 1.2 Mbit/s. Every node has a finite transmit-airtime budget per second, so a constrained radio congests rather than silently absorbing bytes.

Defaults: 30 nodes in 350×350 m (mean degree 4.8, mean connected-pair fraction 0.87), speed 1–5 m/s, 200 s at 1 s resolution, Poisson traffic at 3 messages/s between uniformly random pairs, 200 B payloads, TTL 8, message lifetime 30 s, three link-layer ARQ attempts per hop, and a 5 s link-state refresh. Traffic is generated only in $[10, 170]$ s so that every message gets a full lifetime. We acknowledge the known non-uniform spatial density of random waypoint [27], [28];

all protocols see the identical trace, so the comparison is unaffected. Fig. 1 shows a representative snapshot.

A modelling choice deserves emphasis because it turned out to govern every conclusion in this paper. When a hop exhausts its three ARQ attempts, the packet is *not* lost: the holder still has it, buffers it, and retries on the next step, re-routing if its view has moved on [18]. Only the message lifetime finally discards it. An earlier version of our simulator destroyed the packet on the first ARQ exhaustion, and that single assumption was responsible for 21.5% of all message loss — against 3–4% from route breaks. Under it, link-failure prediction was addressing a term five times smaller than the dominant one, and every predictive result was correspondingly muted. We report this because the same assumption is easy to make silently, and it is not a detail: it decides what a predictive routing experiment is even capable of measuring.

B. Protocols

All five share the same simulator, traffic and link layer. *AODV-like* holds no topology state and floods a route request on demand, paying $2|C|$ request/reply messages for the source's connected component C . It then *caches the discovered route and reuses it until a hop fails*, which is the defining behaviour of a reactive protocol: a large one-off discovery cost followed by a route that grows stale. Re-flooding per packet would give it both an absurd control cost and a permanently fresh view, and would be neither AODV nor a fair baseline. *Shortest-hop* and *OLSR-ETX* are proactive link-state protocols that route on a view refreshed every 5 s — hence up to 5 s stale — by hop count and by ETX [8] respectively. *NEXUS-D* uses (2) and *NEXUS-G* uses (3) on the same view and the same control budget, so any difference between the last three is attributable to the decision rule alone. Every protocol performs link-layer local repair: a failed transmission removes that edge from the local view immediately.

C. Metrics and statistics

Our headline delivery metric is *on-time delivery*: the fraction of generated messages that arrive within a deadline D . This is a deliberate departure from the usual practice of reporting a single PDR, and Section VIII-A shows why it is necessary. A correctly buffered store-and-forward mesh eventually delivers almost everything, so eventual PDR saturates near 0.95 for every protocol we test and stops separating them — while systematically flattering the protocols that simply retry more. What a deadline-bound application experiences is on-time delivery, and there the protocols differ sharply. We report $D \in \{2, 5, 10\}$ s alongside eventual delivery at the 30 s lifetime, so the reader can see the whole curve rather than one point of it.

We also report mean and 95th-percentile end-to-end latency, hop count, route lifetime (how long a selected route stays intact in the true topology, measured on 24 monitored pairs per run), route breaks, control and data bytes, bytes per delivered message, and an energy proxy. Each configuration runs on 20

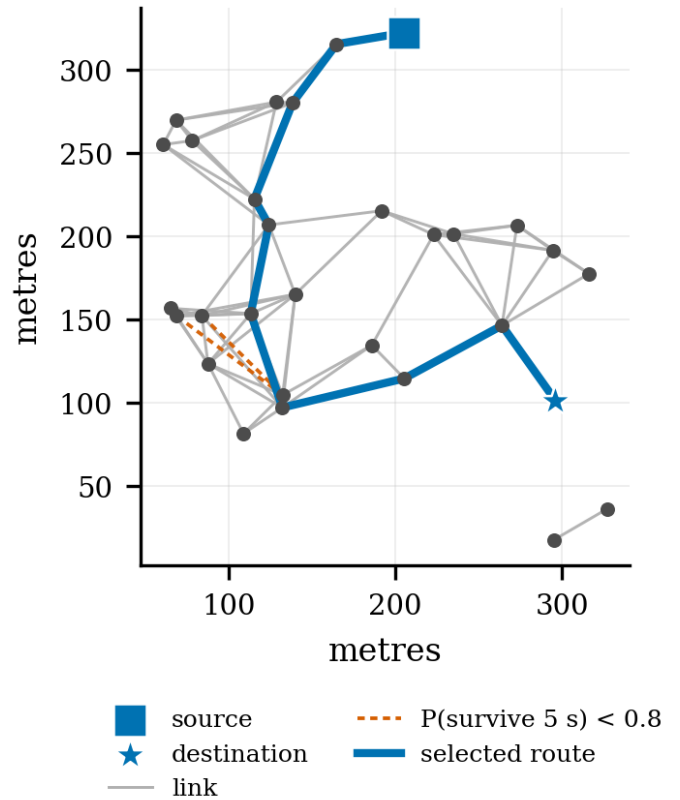


Fig. 1. One snapshot of the simulated mesh (30 nodes, 350×350 m) with the longest route NEXUS-D selects. Dashed edges are those the GNN gives less than 0.8 probability of surviving the next 5 s.

scenario seeds; we report means with 95% confidence intervals and use paired Wilcoxon signed-rank tests across seeds.

D. Hyperparameter fitting

The six weights of (2) and the prediction weight w_{pred} of (3) are fitted by coordinate descent on the 16 scenario seeds 200–215, maximising on-time delivery at $D = 2$ s — the deadline at which routing strategies actually separate, and therefore the only one at which fitting is informative. Those seeds are disjoint from the 20 evaluation seeds and from the GNN's training and validation seeds; no weight is hand-picked and none is refitted on evaluation data. w_{pred} is fitted *per refresh interval* rather than once, because how far a survival prediction should be trusted depends both on how stale the view is and on how accurate the predictor is at that horizon — a single frozen value is not merely suboptimal at other intervals, it is what made prediction actively harmful at long ones in our earlier runs.

VIII. RESULTS

A. Against the classical baselines

Table II and Fig. 2 give the head-to-head, and the first thing to say about it is that two reasonable delivery metrics produce two opposite winners.

Ranked by *eventual* delivery, the order is AODV-like 0.961, shortest-hop 0.958, NEXUS-D 0.938, NEXUS-G 0.930,

TABLE II

ROUTING STRATEGIES AT DEFAULT SETTINGS (30 NODES, 20 SEEDS, $\pm 95\%$ CI). “ON TIME” IS DELIVERY WITHIN A 2 S DEADLINE; “EVENTUAL” IS DELIVERY WITHIN THE 30 S MESSAGE LIFETIME. CONTROL AND DATA BYTES ARE TOTALS PER 200 S RUN.

Strategy	On time (2 s)	Eventual (30 s)	Latency (ms)	p95 (ms)	Hops	Route life (s)	Control (kB)	kB/delivery
AODV-like (reactive)	0.593 ± 0.017	0.961	3297	14204	3.17	1.36	3085	9.97
Shortest-hop (link-state)	0.599 ± 0.016	0.958	3393	15301	3.21	3.86	180	3.51
OLSR-like (ETX)	0.746 ± 0.020	0.921	2181	12602	3.85	6.92	176	2.84
NEXUS-D (multi-metric)	0.736 ± 0.020	0.938	2321	13108	3.68	6.58	176	2.81
NEXUS-G (predictive)	0.757 ± 0.017	0.930	2143	12749	3.84	7.94	176	2.78

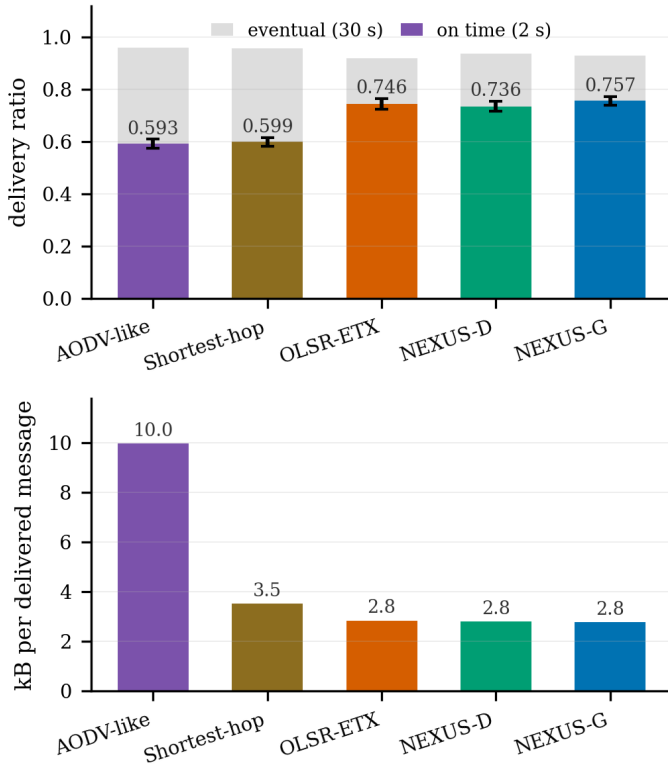


Fig. 2. On-time delivery (coloured) against eventual delivery (grey), and communication cost per delivered message. The grey bars are nearly equal; the coloured ones are not.

OLSR-ETX 0.921: the two strategies with no notion of link quality come first and the loss-aware ones come last. Ranked by delivery within a 2 s deadline the order very nearly inverts: NEXUS-G 0.757, OLSR-ETX 0.746, NEXUS-D 0.736, shortest-hop 0.599, AODV-like 0.593. Fig. 3 shows the crossover directly, at about 9 s.

The mechanism is store-and-forward. A hop-count route runs over weak links, exhausts its ARQ budget and buffers; the packet is not destroyed, it is delayed, and with 30 s to play with it usually arrives. Minimising hops minimises total airtime, so under a sufficiently generous deadline hop count is close to the right objective. A study reporting only PDR at a lifetime this long would conclude that shortest-hop routing is competitive with ETX — and would be reporting a property of its own deadline. We therefore treat the deadline as an experimental variable, not an implementation constant, and this is the single

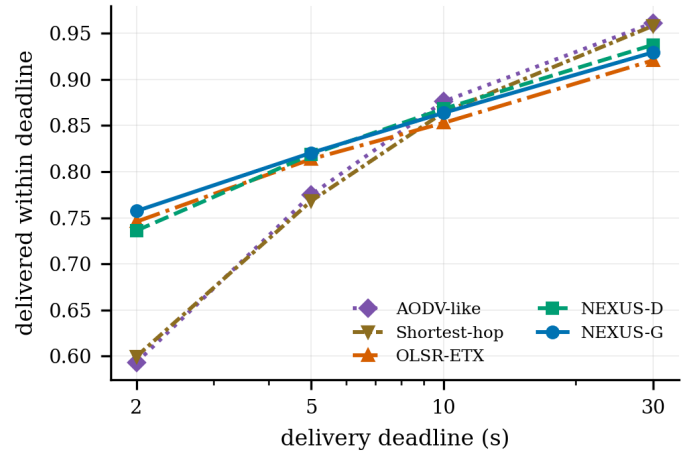


Fig. 3. Delivery against the deadline it is measured at. The curves cross at roughly 9 s: below it, loss-aware routing wins; above it, hop-count routing catches up and passes. A single-number PDR is a point on this figure, and which point decides the ranking.

most consequential methodological point in the paper.

Four paired comparisons carry the section, all on on-time delivery. NEXUS-D beats shortest-hop by +0.137 (+22.8%, $p < 10^{-5}$): pricing links by expected transmissions rather than hop count is worth roughly a fifth of deadline-bound delivery. Against the reactive protocol NEXUS-D uses 71.8% fewer bytes per delivered message and 29.6% less latency (both $p < 10^{-5}$); AODV-like spends 3.1 MB per run rediscovering routes, and its cached routes survive only 1.36 s on average before a hop fails.

The third comparison is the one we least expected. **NEXUS-D does not beat OLSR-ETX** — it loses to it, 0.736 against 0.746 (−1.3%, $p = 0.070$). The full multi-metric cost of (2), with weights fitted on disjoint seeds, is no better than the single classical ETX metric and, on the point estimate, slightly worse. Section VIII-I shows why: several of its terms are individually harmful. We state this plainly because it is the opposite of what a multi-metric routing paper normally reports, and because the tuning procedure was given every opportunity to avoid it.

The fourth is more encouraging: NEXUS-G improves on NEXUS-D by +0.021 (+2.9%, $p = 0.030$), extends route lifetime from 6.58 s to 7.94 s (+21%), and cuts route breaks from 29.1 to 16.6 per run (−43%). It is the only variant that beats plain ETX. Prediction, not the multi-metric sum, is what adds value on top of a good link metric.

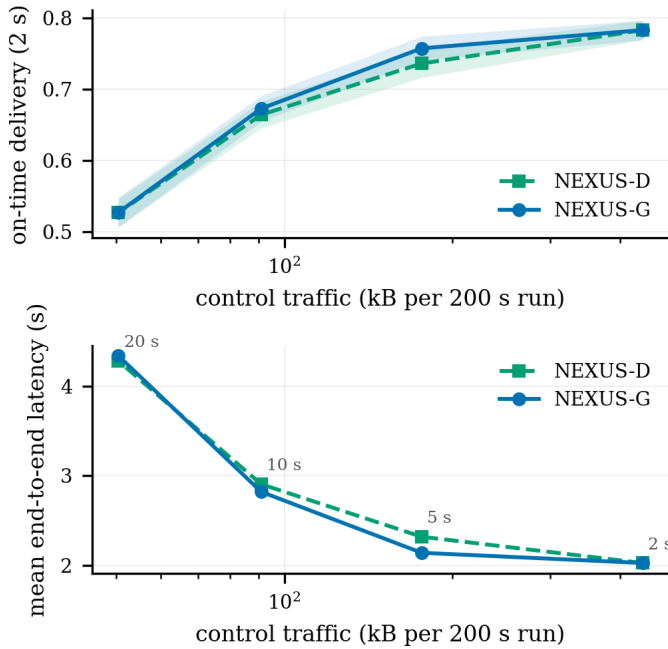


Fig. 4. On-time delivery and latency against control traffic as the link-state refresh interval varies from 2 s (right) to 20 s (left).

B. Prediction versus control overhead

If prediction compensates for a stale view, a network could refresh its topology less often and spend the savings elsewhere. Fig. 4 sweeps the refresh interval from 2 s to 20 s, an $8.8\times$ range in control traffic, with a horizon-matched predictor and an independently fitted w_{pred} at each point.

The answer is that prediction does not substitute for control traffic; it has a narrow operating point and no more. At a 2 s refresh the fitting procedure independently chose $w_{\text{pred}} = 0$ — with a view at most 2 s old there is nothing for a survival prediction to add, and NEXUS-G reduces exactly to NEXUS-D. At 5 s prediction is worth $+2.9\%$ on-time delivery ($p = 0.030$) and -7.7% latency ($p = 0.097$). At 10 s the gain has shrunk to $+1.3\%$ and is no longer significant ($p = 0.177$). At 20 s it is worth nothing measurable ($+0.0\%$, $p = 0.898$).

Meanwhile the cost of a slow control plane is severe and monotone: NEXUS-D falls from 0.783 to 0.527 on-time delivery and from 2.0 s to 4.3 s mean latency across the same sweep. Halving control traffic from 176 kB to 91 kB costs 7 points of on-time delivery, and prediction returns about one of them. The honest summary is that a mesh should refresh its topology often enough to route on, and prediction is a refinement of a healthy control plane rather than a replacement for a starved one. Section VIII-E shows this is a limitation of our predictor at long horizons rather than of the idea.

C. Mobility

We sweep node speed over a $12\times$ range. Route lifetime falls and breaks rise from 8 to 218 per run, so the mechanism prediction targets becomes far more common — but the benefit

does not grow with it, because predictor quality falls at the same time.

At the quasi-static end (0.5–1.5 m/s) prediction is unhelpful (0.762 against NEXUS-D’s 0.766): links are not breaking, so there is nothing to predict. At 1–5 m/s it is at its best (0.757 against 0.736, with breaks cut from 29 to 17). At 3–10 m/s it still leads (0.671 against 0.667, breaks 103 $\rightarrow$ 75). At 6–18 m/s the three strategies converge to within half a point (0.552, 0.551, 0.548) even though breaks have risen to 196–218: when links survive less than the prediction horizon, a 5 s forecast is stale before it is acted on and steering away from doomed links merely selects different doomed links.

Predictive routing therefore has a mobility band, and it is a middle one, bounded below by having nothing to predict and above by not being able to predict it. The graph-level effect is robust — prediction suppresses route breaks at every speed we tested — and only its conversion into delivery is speed-dependent.

D. Scale

Holding node density constant, Fig. 5 varies the network from 10 to 80 nodes. On-time delivery declines for every strategy as paths lengthen and more hops mean more chances to stall. The reactive protocol degrades worst: its per-route flooding cost grows with component size, and it falls from 0.771 to 0.374 across the sweep while its byte cost per delivered message grows to several times NEXUS-D’s.

The interesting structure is at the two ends. At 10 nodes plain OLSR-ETX is best (0.847), ahead of NEXUS-G (0.839) and NEXUS-D (0.826): short paths leave little for either the extra cost terms or prediction to exploit. At 80 nodes the ordering reverses and NEXUS-D leads (0.550 against 0.515 for OLSR-ETX and 0.528 for NEXUS-G), the one regime where the multi-metric terms that Section VIII-I finds harmful at 30 nodes visibly help: with long paths, avoiding low-battery and low-reliability relays begins to pay. That NEXUS-G falls behind NEXUS-D at 80 nodes is consistent with the same reading — w_{pred} was fitted at 30 nodes, and the balance between prediction and resource terms is evidently size-dependent. We did not refit per size, and flag this as a limitation rather than tuning it away.

E. The oracle bound on prediction

Section VIII-A showed prediction is worth $+2.9\%$ on-time delivery. The question that matters for the research programme is not whether that is positive but how much room is left above it. We answer it by replacing the GNN with an oracle that knows exactly which links survive the next interval (Table III); no predictor, however good, can beat it.

At the 5 s operating point the oracle reaches 0.762 against the GNN’s 0.757 and 0.736 with no prediction. The total achievable gain from link-survival prediction is therefore 2.6 points, and our 5,089-parameter model captures 83% of it. That is a tight bound in both directions: it says the GNN is close to done — a larger or better-trained model has at most

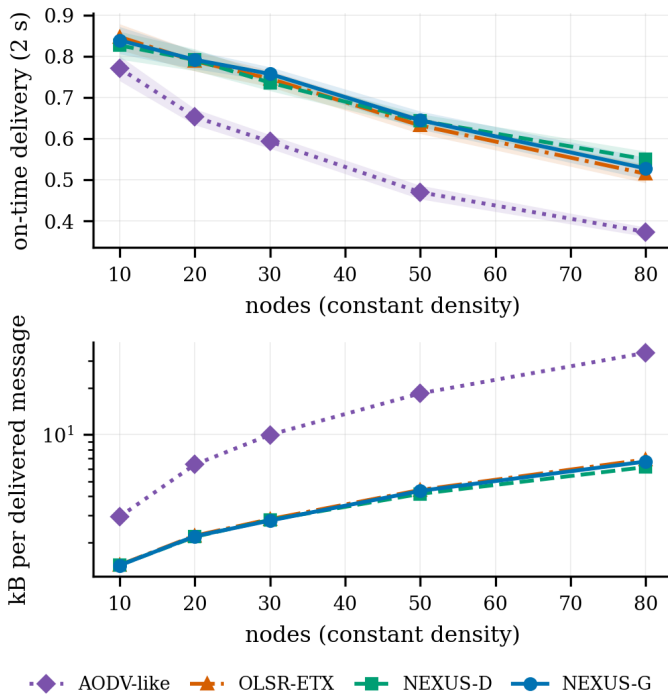


Fig. 5. On-time delivery and per-delivery byte cost against network size at constant node density.

TABLE III

TOP: LINK-SURVIVAL PREDICTION QUALITY ON HELD-OUT TRACES; “BASE” IS THE RATE OF THE MAJORITY (SURVIVES) CLASS. BOTTOM: THE RESULTING ROUTING PERFORMANCE, WITH A PERFECT-KNOWLEDGE ORACLE AS THE UPPER BOUND. ON-TIME DELIVERY AT A 2 S DEADLINE, 20 SEEDS.

Predictor quality (5,089 params, 19.9 kB, 0.49 ms/snapshot)					
Horizon	Samples	AUROC	Accuracy	Base	Brier
5 s	29,494	0.953	0.930	0.865	0.052
10 s	14,316	0.852	0.826	0.743	0.125
20 s	6,724	0.689	0.641	0.529	0.222
Routing performance					
Refresh	Predictor	On time	Lat. (ms)	Route life (s)	Breaks
5 s	none	0.736	2321	6.58	29.1
	GNN	0.757	2143	7.94	16.6
	oracle	0.762	2140	8.13	12.8
20 s	none	0.527	4278	2.81	151.6
	GNN	0.527	4337	3.05	149.8
	oracle	0.558	3969	4.48	74.7

half a point left to win — and it says the mechanism is real rather than an artifact of a particular predictor.

The 20 s row is the more informative one, and it is a negative result about our model rather than about prediction. There the GNN is worth nothing at all (0.527 against 0.527), while the oracle gains 3.1 points and halves route breaks. The ceiling has not disappeared; our predictor has simply stopped being able to reach it, at AUROC 0.689 on a 20 s horizon against a base rate of 0.529. Longer horizons are where the remaining research headroom lies, and also where the learning problem is hardest: the same architecture that is near-oracle at 5 s is

TABLE IV

BEHAVIOURAL TRUST UNDER PACKET-DROPPING RELAYS, 20 SEEDS, ON-TIME DELIVERY AT A 2 S DEADLINE. Δ AND p COMPARE NEXUS-T AGAINST NEXUS-D ON PAIRED SEEDS. TPR/FPR ARE AT A FIXED 0.5 TRUST THRESHOLD.

Atk.	none	+tr.	+tr.+G	Δ	p	TPR	FPR
0 %	0.736	0.733	0.752	−0.5%	0.701	—	0.020
10 %	0.620	0.674	0.680	+8.8%	$2e^{-5}$	0.97	0.026
20 %	0.512	0.605	0.617	+18.1%	$2e^{-6}$	0.96	0.008
30 %	0.411	0.511	0.507	+24.3%	$2e^{-6}$	0.95	0.010

near-useless at 20 s.

Two design consequences follow. First, prediction should be evaluated against an oracle as a matter of course; without the 0.762 column, our +2.9% would look like a small gain rather than most of what is available. Second, route stability and delivery are only partly coupled: the oracle more than halves route breaks at 5 s (29.1 $\rightarrow$ 12.8) to buy 2.6 points of on-time delivery, because with link-layer local repair a break costs a retry interval of latency rather than the message. A system seeking large delivery gains from graph learning should predict something else — end-to-end path viability, congestion, or partition timing — or target a setting where repair is expensive, such as connection-oriented transports where a break restarts a transfer.

F. Trust against packet-dropping relays

We place a fraction of nodes under adversarial control; each silently drops 80% of the transit traffic it is asked to forward, while behaving normally otherwise. Trust is derived only from watchdog observations [14], never from an oracle, and enters routing through the w_{distrust} term of (2).

Table IV and Fig. 6 show the outcome. This is the component that pays, and it pays more the worse things get: +8.8%, +18.1% and +24.3% on-time delivery at 10%, 20% and 30% attacker density, all significant at $p < 10^{-4}$. Detection is accurate and stable across densities — true-positive rate 0.95–0.97 at a fixed 0.5 threshold, with a false-positive rate never above 0.026.

Two details are worth stating because they cut against a clean story. First, the false-positive rate is not zero even with *no* attackers present (0.020): a watchdog cannot distinguish a relay that dropped a packet from a relay whose next hop exhausted its ARQ budget, so a bad link is indistinguishable from bad behaviour. This is intrinsic to promiscuous observation, not an artifact of our threshold. Second, that misattribution is close to free here — trust costs 0.5% on-time delivery in a clean network and the difference is not significant ($p = 0.701$), so the EWMA floor and slow recovery rate are doing their job of preventing an unlucky observation from becoming a persistent exclusion.

Combining trust with prediction (the +trust+GNN column) helps at low attacker densities and stops helping at 30%, where it falls slightly below trust alone (0.507 against 0.511). Under heavy attack the binding constraint is which relays are honest, not which links will survive.

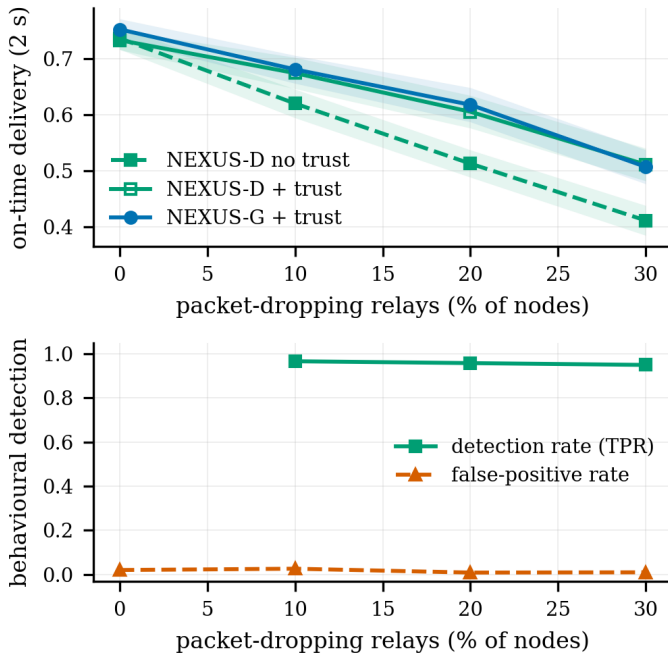


Fig. 6. On-time delivery against attacker density, with and without the behavioural trust graph.

TABLE V
BEHAVIOURAL TRUST AGAINST ADAPTIVE ATTACKERS, 20% ATTACKER DENSITY, 20 SEEDS, ON-TIME DELIVERY AT A 2 S DEADLINE. Δ AND p COMPARE NEXUS-T AGAINST NEXUS-D ON PAIRED SEEDS. “DROPS” IS PACKETS DISCARDED BY ATTACKERS PER RUN, WITHOUT / WITH TRUST.

Attacker	none	+tr.	Δ	p	TPR	FPR	Drops
static 80%	0.512	0.605	+18.1%	$2e^{-6}$	0.96	0.008	176/106
grey hole 30%	0.652	0.671	+2.9%	0.012	0.61	0.012	76/53
on-off	0.637	0.671	+5.3%	$6e^{-5}$	0.64	0.017	88/58
evasive	0.512	0.678	+32.3%	$2e^{-6}$	0.27	0.025	176/43

G. Trust against adaptive attackers

The attackers above do not react to being distrusted. We therefore also test, at 20% density and with trust unchanged, a *grey hole* dropping only 30% of transit traffic, an *on-off* attacker alternating 20 s of 80% dropping with 20 s of honest forwarding, and an *evasive* attacker that reads its own trust score and drops only while it stays above 0.65, so it never crosses the detection threshold.

Table V shows that adaptive attackers defeat *detection* (TPR falls to 0.27 for the evasive attacker) but not *protection*: because trust enters routing as a continuous cost rather than a blacklist, it still improves on-time delivery significantly in every case. To stay undetected the evasive attacker must forward most traffic, so its drops fall from 176 to 43 per run. The weakest case is the low-rate grey hole (+2.9%), which is hard to separate from a relay with a poor link.

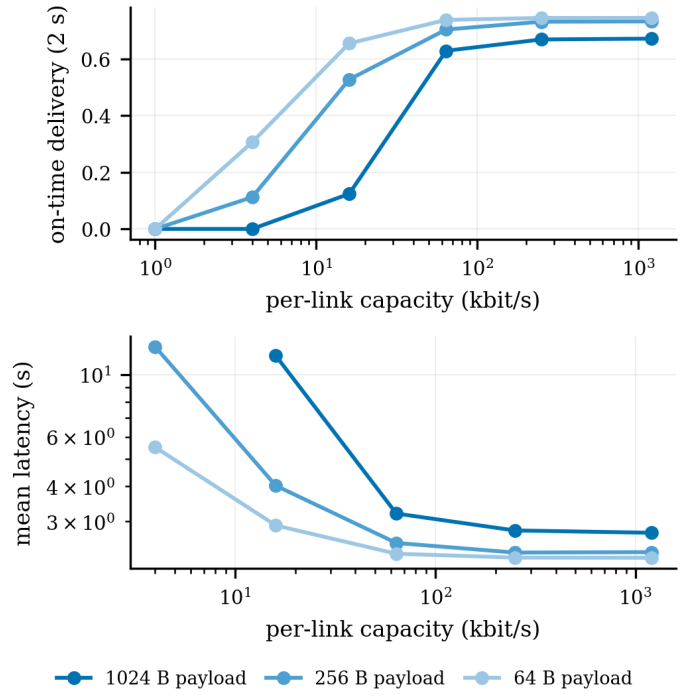


Fig. 7. On-time delivery and latency against per-link capacity for three payload sizes. Each series ends where (4) predicts it must.

H. Ultra-low bandwidth

With a fixed per-node airtime budget A and per-link capacity C , a packet carrying P payload bytes is transmissible only if

$$\frac{8(H + K + P)}{C} \leq A, \quad (4)$$

where $H = 34\text{B}$ is the header and $K = 124\text{B}$ the cryptographic expansion. With $A = 1\text{s}$ this predicts total collapse below $C = 8(158 + P)\text{ bit/s}$: 1776 bit/s for a 64-byte message, 3312 for 256 bytes and 9456 for 1024 bytes. Fig. 7 confirms all three thresholds exactly. At 4 kbit/s the 1024-byte configuration delivers nothing at all — it is below its 9456 bit/s threshold — while 256- and 64-byte messages still work; at 1 kbit/s every payload size is below threshold and nothing gets through.

This sweep also gives the sharpest illustration of why the deadline matters. Take the 1024-byte payload as capacity falls from 1.2 Mbit/s to 16 kbit/s: *eventual* delivery moves only from 0.940 to 0.883, a network that by that measure is still working. On-time delivery over the same range collapses from 0.672 to 0.124, and mean latency rises from 2.7 s to 11.7 s. The messages are not being lost; they are being deferred, queued behind a radio that can no longer clear them, and a lifetime-bounded PDR counts a message that arrives 11 s late as a success. At a 64-byte payload the same squeeze is far gentler ($0.745 \rightarrow 0.655$ on time), which is the practical recommendation: under airtime pressure, shrink the message.

Two consequences follow. First, the 158-byte fixed cost dominates the frontier for small messages: at a 64-byte payload it is 71% of the packet, so amortising the ephemeral key

TABLE VI

COST-FUNCTION ABLATION, 20 SEEDS, ON-TIME DELIVERY AT A 2 S DEADLINE. THE LOSS-LINEAR VARIANT SUBSTITUTES A PENALTY OF EQUAL MAGNITUDE TO THE CONVEX ETX TERM IT REPLACES, SO THE ROW ISOLATES CONVEXITY RATHER THAN WEIGHT. THE LAST TWO ROWS USE A SCENARIO WITH 20% PACKET-DROPPING RELAYS, WHERE THE TRUST TERM IS ACTIVE.

Variant	On time	Event.	Lat. (ms)	Hops
full	0.736 ± 0.020	0.938	2321	3.68
– energy	0.742 ± 0.017	0.940	2299	3.64
– reliability	0.733 ± 0.016	0.950	2438	3.54
– battery	0.751 ± 0.019	0.935	2212	3.78
– latency	0.741 ± 0.020	0.933	2267	3.77
ETX term only	0.745 ± 0.016	0.945	2248	3.60
– ETX (linear in loss)	0.699 ± 0.016	0.859	2092	4.32
+ trust (20% malicious)	0.605 ± 0.029	0.750	1936	3.32
– trust (20% malicious)	0.512 ± 0.024	0.613	1753	3.12

exchange and signature across a session — not compressing the payload — is the change that would move (4) materially. Second, reporting delivery ratio alone under congestion is actively misleading, in the specific and checkable sense that it reports 0.883 for a network delivering 12% of its traffic on time.

I. Ablation

Table VI removes one term of (2) at a time, with every other weight held at its fitted value.

The ablation is decisive and it is unflattering to the proposed cost function. **Removing a term usually improves the system.** Dropping the battery term gains 1.5 points ($0.736 \rightarrow 0.751$), dropping energy gains 0.6, dropping the latency term gains 0.5, and keeping *only* the ETX term gains 0.9 — placing plain ETX above the full multi-metric cost, which is exactly what the head-to-head in Section VIII-A reported. Only the node-reliability term is worth anything at all, and only 0.3 points, well inside its confidence interval. The multi-metric apparatus of (2) is not merely decorative in a benign network; several of its terms are actively counterproductive, and the deterministic system would be better if they were deleted.

One term survives, and it is the one the cost was built around. Replacing the convex ETX term with a loss-linear penalty of equal magnitude costs 3.7 points of on-time delivery ($0.736 \rightarrow 0.699$), costs 7.9 points of eventual delivery ($0.938 \rightarrow 0.859$), and inflates hop count from 3.68 to 4.32. Delivery along a path is multiplicative in per-hop success, so a cost that is linear in loss systematically under-prices weak links; that is the single mechanism in the deterministic layer that pays for itself.

The trust term is the other exception, worth 9.3 points under attack ($0.512 \rightarrow 0.605$) — larger than every benign-network term in the table combined, and in the opposite direction to most of them.

We draw the obvious conclusion rather than hedging it. A deterministic router for this regime should carry a convex loss metric and a behavioural trust term, and nothing else; the energy, battery and latency terms should be dropped unless

a deployment has a specific reason to keep them, and adding them because they are individually plausible is how a routing metric gets worse while looking more sophisticated.

IX. DISCUSSION AND LIMITATIONS

What we would tell a designer. Decide your delivery deadline before you choose a routing metric; if you can tolerate tens of seconds, a buffered hop-count protocol is hard to beat and everything below matters less than you think. If you cannot, carry a convex loss metric — it is worth 3.7 points and costs nothing — and add behavioural trust if misbehaviour is in your threat model, since it is close to free when you are wrong about that. Do not add energy, battery or latency terms to the cost because they sound reasonable: each of them made our system measurably worse, and the combination lost to a metric from 2003. Add link-survival prediction if your control plane refreshes on the order of the link lifetime, where it is worth about 3% and most of what an oracle could give; not if your view is nearly fresh, where the fitting procedure sets its weight to zero, and not if it is badly stale, where our predictor stops working.

Simulation only. Every number here comes from our simulator. The radio model is a standard log-distance path loss with AR(1) shadowing [26], not a measured channel; there is no interference, capture effect, or MAC contention model beyond a per-node airtime budget; mobility is random waypoint with its known spatial bias [27], [28]. Absolute delivery ratios should not be compared to field measurements. The relative comparisons are on firmer ground, since all protocols see byte-identical traces.

Idealised control plane. Our proactive protocols share a single link-state view rather than maintaining per-node views with independent staleness, and the trust graph is likewise a shared consensus rather than per-node opinion with second-hand reputation exchange. This flatters proactive protocols relative to the reactive baseline and understates the difficulty of distributing trust. A per-node view is the most important next change to the simulator.

Expected behaviour in a real deployment. Real radios add MAC contention, interference, quantised and noisy RSSI, and less regular human mobility, and each node would hold its own, differently stale view. We expect this to delay the deadline crossover (more queueing delay) and to shrink, but not reverse, the advantage of the convex loss metric, since ETX has been validated on testbeds [8]. Link-survival prediction is the most fragile result: noisier RSSI trends and per-node views make the task harder, so the 83% oracle capture would likely fall and the model would need retraining on measured traces. Watchdogs would also miss transmissions (hidden terminals, collisions), raising trust's false-positive rate. We therefore expect the relative orderings to transfer better than the absolute gains.

One attack family, one predictor. We model packet-dropping relays only, in static and adaptive forms (Section VIII-G). Selective forwarding by destination, sinkhole, collusion and Sybil behaviour [16], [17] are not evaluated;

Sybil in particular is a cryptographic identity problem our behavioural layer cannot address. The GNN predicts link survival only — node-risk and route-quality heads are specified in our architecture but not evaluated here, and the oracle result suggests link survival was the least promising of the three to start with.

Scope of the trust evaluation. Detection rates are computed at a fixed 0.5 threshold rather than swept into an ROC curve. The adaptive attackers of Section VIII-G follow fixed rules; an attacker that optimises its drop schedule against the routing cost could do more harm.

X. CONCLUSION

We built an infrastructure-free mesh communication system with a learned link-survival predictor and a behavioural trust graph, and measured what each component contributes rather than only what the whole system achieves. The result is mostly a set of negative and bounded findings, which we think is the useful kind.

The delivery metric decides the ranking: eventual delivery saturates and flatters hop-count routing, on-time delivery inverts the order, and the two cross at about 9 s. The deterministic multi-metric cost — the conventional part of the design — does not beat plain ETX, and three of its terms are individually harmful; only convexity in packet loss survives its own ablation. Link-survival prediction does pay, modestly and within a narrow band, and an oracle shows our 19.9 kB model already captures 83% of the available gain at a 5 s horizon while capturing none of it at 20 s. Behavioural trust is the component that pays outright, recovering up to 24.3% of on-time delivery under attack at no significant cost when there is no attacker.

Two methodological points transfer beyond this system. The first is the oracle comparison: an end-to-end improvement makes a learned component easy to justify, but bounding what it could contribute if it were perfect is what tells you whether to keep working on it — here it said the 5 s model is finished and the 20 s one is not. The second is that a single delivery number is a point on a deadline curve, and which point you pick can reverse your conclusion. We report the curve.

REFERENCES

- [1] C. E. Perkins and E. M. Royer, “Ad-hoc on-demand distance vector routing,” in *Proc. 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA)*, 1999, pp. 90–100.
- [2] T. Clausen and P. Jacquet, “Optimized link state routing protocol (OLSR),” IETF, RFC 3626, 2003.
- [3] D. B. Johnson and D. A. Maltz, “Dynamic source routing in ad hoc wireless networks,” in *Mobile Computing*. Kluwer, 1996, pp. 153–181.
- [4] K. Rusek, J. Suárez-Varela, P. Almasan, P. Barlet-Ros, and A. Cabellos-Aparicio, “RouteNet: Leveraging graph neural networks for network modeling and optimization in SDN,” *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 10, pp. 2260–2270, 2020.
- [5] P. Almasan, J. Suárez-Varela, K. Rusek, P. Barlet-Ros, and A. Cabellos-Aparicio, “Deep reinforcement learning meets graph neural networks: Exploring a routing optimization use case,” *Computer Communications*, vol. 196, pp. 184–194, 2022.
- [6] F. Geyer and G. Carle, “Learning and generating distributed routing protocols using graph-based deep learning,” in *Proc. ACM SIGCOMM Workshop on Big Data Analytics and Machine Learning for Data Communication Networks (Big-DAMA)*, 2018, pp. 40–45.
- [7] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva, “A performance comparison of multi-hop wireless ad hoc network routing protocols,” in *Proc. 4th ACM MobiCom*, 1998, pp. 85–97.
- [8] D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris, “A high-throughput path metric for multi-hop wireless routing,” in *Proc. 9th ACM MobiCom*, 2003, pp. 134–146.
- [9] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J.-P. Vasseur, and R. Alexander, “RPL: IPv6 routing protocol for low-power and lossy networks,” IETF, RFC 6550, 2012.
- [10] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” in *Proc. Int. Conf. on Learning Representations (ICLR)*, 2017.
- [11] W. L. Hamilton, R. Ying, and J. Leskovec, “Inductive representation learning on large graphs,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017, pp. 1024–1034.
- [12] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, “Graph attention networks,” in *Proc. Int. Conf. on Learning Representations (ICLR)*, 2018.
- [13] E. Rossi, B. Chamberlain, F. Frasca, D. Eynard, F. Monti, and M. Bronstein, “Temporal graph networks for deep learning on dynamic graphs,” in *ICML Workshop on Graph Representation Learning*, 2020.
- [14] S. Marti, T. J. Giuli, K. Lai, and M. Baker, “Mitigating routing misbehavior in mobile ad hoc networks,” in *Proc. 6th ACM MobiCom*, 2000, pp. 255–265.
- [15] S. Buchegger and J.-Y. Le Boudec, “Performance analysis of the CONFIDANT protocol: Cooperation of nodes — fairness in dynamic ad-hoc networks,” in *Proc. 3rd ACM MobiHoc*, 2002, pp. 226–236.
- [16] C. Karlof and D. Wagner, “Secure routing in wireless sensor networks: Attacks and countermeasures,” *Ad Hoc Networks*, vol. 1, no. 2–3, pp. 293–315, 2003.
- [17] J. R. Douceur, “The sybil attack,” in *Proc. 1st Int. Workshop on Peer-to-Peer Systems (IPTPS)*, 2002, pp. 251–260.
- [18] K. Fall, “A delay-tolerant network architecture for challenged internets,” in *Proc. ACM SIGCOMM*, 2003, pp. 27–34.
- [19] A. Vahdat and D. Becker, “Epidemic routing for partially connected ad hoc networks,” Duke University, Tech. Rep. CS-2000-06, 2000.
- [20] A. Langley, M. Hamburg, and S. Turner, “Elliptic curves for security,” IETF, RFC 7748, 2016.
- [21] D. J. Bernstein, “Curve25519: New Diffie-Hellman speed records,” in *Public Key Cryptography (PKC)*, 2006, pp. 207–228.
- [22] H. Krawczyk and P. Eronen, “HMAC-based extract-and-expand key derivation function (HKDF),” IETF, RFC 5869, 2010.
- [23] Y. Nir and A. Langley, “ChaCha20 and Poly1305 for IETF protocols,” IETF, RFC 8439, 2018.
- [24] S. Josefsson and I. Liusvaara, “Edwards-curve digital signature algorithm (EdDSA),” IETF, RFC 8032, 2017.
- [25] T. Camp, J. Boleng, and V. Davies, “A survey of mobility models for ad hoc network research,” *Wireless Communications and Mobile Computing*, vol. 2, no. 5, pp. 483–502, 2002.
- [26] T. S. Rappaport, *Wireless Communications: Principles and Practice*, 2nd ed. Prentice Hall, 2002.
- [27] C. Bettstetter, G. Resta, and P. Santi, “The node distribution of the random waypoint mobility model for wireless ad hoc networks,” *IEEE Transactions on Mobile Computing*, vol. 2, no. 3, pp. 257–269, 2003.
- [28] J. Yoon, M. Liu, and B. Noble, “Random waypoint considered harmful,” in *Proc. IEEE INFOCOM*, 2003, pp. 1312–1321.