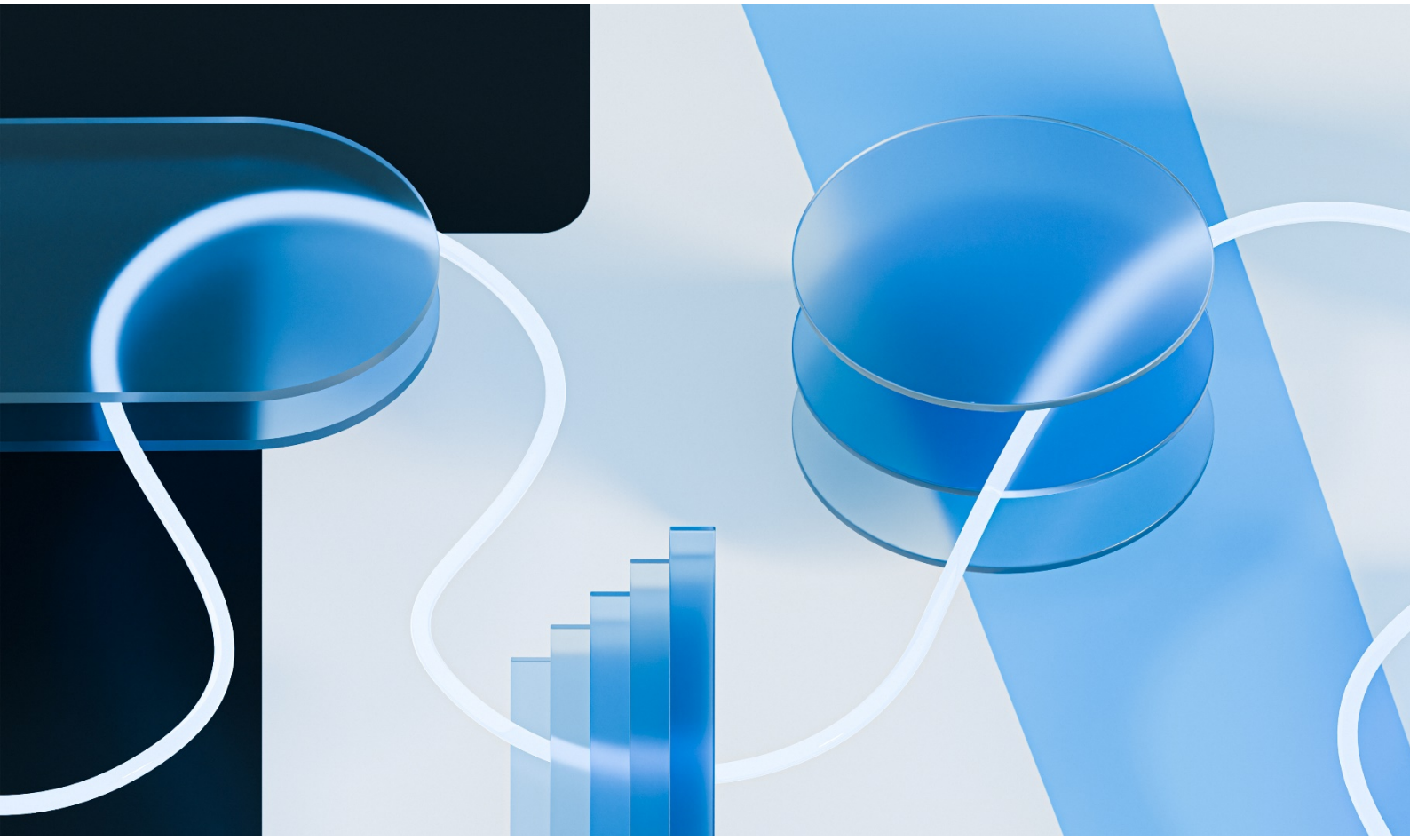
twon
twin of online social networks



TWin of Online Social Networks

Deliverable D2.2

General Report on Simulation Model

Main Authors: Michael Mäs and Fabio Sartori

Funded by
the European Union

About TWON

TWON (project number 101095095) is a research project, fully funded by the European Union, under the Horizon Europe framework (HORIZON-CL2-2022-DEMOCRACY-01, topic 07). TWON started on 1 April 2023 and will run until 31 March 2026. The project is coordinated by the Universiteit van Amsterdam (the Netherlands) and implemented together with partners from Universität Trier (Germany), Institut Jozef Stefan (Slovenia), FZI Forschungszentrum Informatik (Germany), Karlsruher Institut für Technologie (Germany), Robert Koch Institute (Germany), Univerzitet u Begogradu - Institut za Filozofiju I Drustvenu (Serbia) and Slovenska Tiskovna Agencija (Slovenia), Dialogue Perspectives e.V (Germany).

Funded by the European Union. Views and opinions expressed are those of the author(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them.

Funded by the European Union

Project Name	Twin of Online Social Networks
Project Acronym	TWON
Project Number	101095095
Deliverable Number	D2.2
Deliverable Name	General Report on Simulation Model
Due Date	31.03.2026
Submission Date	27.03.2026
Type	R - Document/ Report
Dissemination Level	PU - Public
Work Package	WP 2
Lead beneficiary	5-KIT
Contributing beneficiaries and associated partners	Universiteit van Amsterdam (UvA), Universität Trier (UT), Institut Jozef Stefan (JSI), FZI Forschungszentrum Informatik (Germany), Karlsruher Institut für Technologie (KIT), Robert Koch Institut (RKI), Univerzitet u Begogradu - Institut za Filozofiju I Drustvenu (UoB), Slovenska Tiskovna Agencija (STA), Dialogue Perspectives e.V (DIA)

Executive Summary

Many caution that online social networks contribute to undesirable social dynamics such as opinion polarization, the spread of fake news, conspiracy theories, discrimination, and large-scale collective outrage. Although these phenomena are well documented in the scientific literature, demonstrating that online social networks have contributed to their emergence has proven elusive. Digital twins of online social networks, TWONs, hold the promise of addressing this problem. These highly advanced and realistic computer models enable the quantification of the extent to which online social networks, as well as specific algorithm design choices, yield undesirable outcomes. Furthermore, they offer a means to optimize the design of online social networks with respect to social, ethical, and epistemic objectives. Accordingly, TWONs might be a powerful means to regulate the design of online social networks.

In the present document, the TWON consortium is describing models developed and analyzed in the project. In Deliverable 2.1, we have detailed the ingredients of a TWON in very general terms. Here, we show of this general framework has been tranlated into four versions of the TWON, each developed for a specific purpose and tailored to a specific context.

twon
twin of online social networks

Contents

List of Abbreviations

LLM Large Language Model

OSN Online Social Network

TWON Twin of an Online Social Network

Report on Simulation Model

Michael Mäs, Fabio Sartori *

March 27, 2026

1 How to Build a TWON: From Research Questions to Model Design

Digital Twins of Online Social Networks (TWONs) are not general-purpose simulations, but purpose-driven instruments designed to answer specific research questions about the dynamics of online communication. Their primary value lies in enabling counterfactual analysis, the evaluation of platform design choices, and the assessment of systemic risks in online social networks. As such, the design of a TWON should not only be approached as the construction of a comprehensive or maximally realistic model, but rather as the development of a targeted representation tailored to a clearly defined analytical objective.

A central principle guiding the construction of a TWON is therefore the following: *the research question determines the model*. Before specifying agents, algorithms, or technical infrastructure, it is necessary to clarify what phenomenon one aims to explain, predict, or manipulate and what context this relevant. Different questions require different levels of abstraction, different mechanisms, and different types of data. For instance, questions concerning the spread of misinformation or the emergence of polarization may require detailed modeling of social influence and network structure, and large simulated networks, while questions related to discourse quality or toxicity may necessitate realistic content generation, potentially involving large language models (LLMs).

This leads to a fundamental trade-off in TWON design: the choice of modeling granularity. Highly detailed models, such as those incorporating LLM-based agents, can capture complex and realistic communication patterns but come at substantial computational cost and reduced interpretability. In contrast, more abstract formal models allow for large-scale simulations and clearer identification of causal

*We would like to thank Achim Rettinger and Alenka Guček for reviewing the internal draft of the paper. The report has benefited considerably from the comments.

mechanisms, but rely on stronger simplifying assumptions about user behavior and content. Selecting an appropriate level of granularity is therefore not a purely technical decision, but a methodological one that depends directly on the research question.

Given these considerations, building a TWON can be understood as a structured process that proceeds from (i) defining the research question, to (ii) identifying the relevant mechanisms, (iii) selecting an appropriate level of abstraction, and (iv) implementing the corresponding user and platform models. This section outlines such a process and provides a methodological framework for constructing TWONs that are both sufficiently realistic to yield credible insights and sufficiently flexible to support counterfactual analysis.

Rather than presenting a fixed model, the goal is to provide a set of principles and guidelines that enable researchers to design TWONs adapted to their specific analytical needs, while maintaining transparency about modeling assumptions and their implications.

In Deliverable 2.1 (Prototype), the TWON consortium provided a detailed description of a generalized TWON. This model encompasses all aspects of platform design and user behavior that our work identified as important to represent and proposes a flexible approach to implementation. How and whether each of these aspects is implemented in a given TWON, which parameter values are chosen, and whether aspects are represented through explicit equations or through data- or machine-learned patterns depends on the context of the study (e.g. which OSN, which issue) and the specific research question. In the following, we describe how a TWON tailored to specific settings is developed.

1.1 Defining the Research Question

The first and most consequential step in building a TWON is not choosing a modeling framework, nor selecting a dataset. It is formulating a precise research question. Everything that follows — the level of abstraction, the mechanisms included, the outcomes measured — flows from this choice.

This section describes what a well-formed TWON research question looks like, what types of questions are most amenable to TWON-based investigation, and how to map from a question to the observables and mechanisms that will drive the model design.

1.1.1 What kind of question is a TWON designed to answer?

TWONs are instruments for *counterfactual analysis*. Their comparative advantage over observational or experimental approaches is the ability to manipulate platform design choices that cannot be changed in the real world — ranking algorithms, content visibility rules, notification regimes, network topology — and to observe the downstream effects on social dynamics.

This shapes the class of questions a TWON is well-suited to address. Three types are most common:

- **Causal questions.** Does a given platform mechanism cause or amplify a social phenomenon? For example: does a similarity-based ranking algorithm sustain opinion polarization that would otherwise decay, as the filter-bubble argument suggests?(Keijzer and Mäs, 2022; Pariser, 2011) Does success-driven user activity contribute to the fragmentation of a network into opinion clusters?(Horn et al., in press) These questions require a counterfactual: a simulation in which the mechanism of interest is absent or modified and that serves as a comparison to simulations that implement the mechanism.
- **Design questions.** Which platform configuration produces better outcomes along a specified dimension? For example: does chronological ranking produce more ideologically diverse exposure than engagement-based ranking? Does limiting resharing reduce the spread of low-credibility content? These questions require comparing multiple platform configurations under otherwise identical conditions.
- **Risk assessment questions.** Under what conditions does a social network become susceptible to a given pathology — polarization, information cascades, coordinated inauthentic behavior? These questions typically require sweeping a parameter space across many simulation runs to map the boundary between stable and unstable dynamics.

A question that does not fit any of these categories — for example, a question that is purely descriptive or that concerns individual-level psychological processes without reference to platform mechanisms — is unlikely to benefit from a TWON. In such cases, other methods (surveys, experiments, natural language analysis) are more appropriate, unless there is no access to real-world data and only theoretical analyses are feasible.

1.1.2 From question to observables

A research question is not yet a model specification. The next step is to identify the *observables* that the question implies: what quantities need to be measured in the simulation in order for the question to be answerable?

This step is non-trivial because many theoretical constructs of interest — polarization, discourse quality, filter bubbles, misinformation spread — can be operationalized in multiple ways, each with different implications for model design. The choice of operationalization should be made explicitly and before model construction begins.

Consider polarization. At the population level, it can be operationalized as the variance of an opinion distribution, the degree of network homophily, or the strength of affective in-group/out-group distinctions. Each operationalization implies different model requirements. Variance-based polarization requires a continuous opinion variable and a mechanism of opinion influence. Network homophily requires an explicit social graph. Affective polarization requires agents to represent group identities, not just opinions.

Similarly, discourse quality can be operationalized as the diversity of ideological viewpoints to which a user is exposed, the prevalence of substantive argumentation relative to incivility, or the presence of cross-cutting content. Each of these requires different content representations and different platform mechanisms to be modeled.

The rule is simple: *what cannot be measured in the simulation cannot be used to answer the research question*. Specifying observables first prevents the common failure mode of building a complex model and then discovering that it does not actually produce the data needed to evaluate the hypothesis.

For more on metrics: Selecting the right observables is closely tied to how outcomes are measured. A group of collaborators has developed a set of socially grounded debate quality metrics — covering exposure, diversity, quality, and incivility of content — with detailed operationalizations and validation. See Deliverables D5.1 and D5.2.

1.1.3 From observables to mechanisms

Once the observables are fixed, the question becomes: which social and platform mechanisms are necessary to produce variation in those observables? This is the link between the research question and the model architecture. Not all mechanisms need to be included. A TWON is not a census of everything that happens on a social network. It is a representation of all processes that are *causally relevant* to the outcome of interest. Including irrelevant mechanisms adds noise, increases computational cost, and makes causal interpretation harder.

A useful heuristic is to ask: could the outcome of interest emerge in a model that does *not* include this mechanism? If yes, the mechanism is a candidate for exclusion. If no, it is a candidate for inclusion. This question is best answered not by intuition alone, but by running simple toy models — a practice described in detail in Section 3.

For example, in the study of opinion polarization, it is not obvious a priori whether heterogeneity in user activity matters. One might assume that polarization is driven primarily by opinion dynamics and network structure, and that all users participating equally is a reasonable simplification. However, empirical analysis and toy-model experiments have shown that activity driven by social success — where

users who receive positive feedback become more active, exerting disproportionate influence on their network neighborhood — can produce qualitatively different polarization dynamics than a model in which activity is uniform Horn et al. (in press). A TWON designed to study polarization that omits this mechanism may reach incorrect conclusions. The output of this step is a list of mechanisms that the model must include, with a brief justification for each. This list determines the minimal model architecture.

1.1.4 A note on scope

It is worth being explicit about what a TWON research question is *not*. It is not a question about a specific platform or a specific empirical event. TWONs do not simulate Twitter or Reddit or TikTok. They simulate abstract platforms with specified properties. The connection to real platforms is indirect: platform design choices observed in the real world are used to motivate the parameter values or the mechanisms under study, but the TWON itself is a formal model, not a replica.

This distinction matters because it constrains the kinds of claims a TWON can support. A TWON can support claims of the form: "a platform with ranking algorithm X produces higher polarization than a platform with ranking algorithm Y, under these specified conditions." It cannot support the claim: "this is what happened on Twitter in 2020." Keeping this distinction clear at the outset prevents overinterpretation of results and keeps the research question properly scoped.

1.2 Build the Platform model

As detailed in Deliverable 2.1, a TWON consists of three main ingredients: technical infrastructure, a user model, and the platform model (see Figure 1).

The platform model comprises three distinct components, each corresponding to a different direction of interaction between user and platform. The first are **affordances**: everything that has to do with the user giving input to the platform. The second is **ranking and feed**: everything the platform does to process that input and deliver content back to the user while online. The third is the **notification system**: how the platform interacts with the user while offline.

In conjunction, these three components define the environment within which agents operate. In social science terms, the platform represents the *rules of the game* — the institutional constraints that shape what actions are possible, what information is available, and when users are drawn back in (North, 1990).

Before specifying any agent behavior, the platform must be defined, because it determines the action space available to agents and the information environment they operate in. Platform assumptions

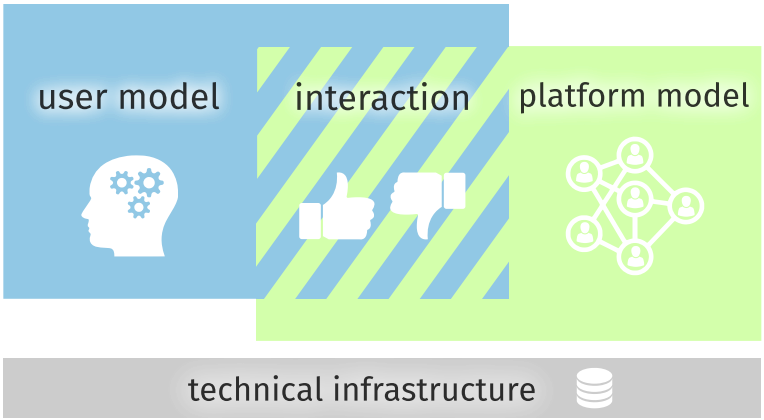


Figure 1: Basic ingredients of a TWON

are therefore *upstream* of agent assumptions.

1.2.1 Affordances

Affordances are the set of actions the platform makes available to the user — everything the user can give as input to the platform. They define what agents *can* do, not what they *will* do. Typical affordances in an online social network include:

- **Content creation**: posting, replying, commenting
- **Reactions**: liking, disliking, sharing/forwarding
- **Network actions**: following, blocking, muting
- **Constraints**: character limits, posting frequency caps

Each affordance that is included becomes a potential action in the agent's decision problem. Each affordance that is *excluded* removes a mechanism from the simulation entirely. The presence or absence of a dislike button, for instance, changes what social feedback signals are available and how agents can express disagreement.

Design Recommendation: Include only the affordances required by the mechanisms identified in your research question and that are present in the context of your study. Every additional affordance adds a corresponding decision to the agent model.

1.2.2 Backend Logic

Backend logic comprises everything the platform does internally, without direct user input and without direct user visibility. It is a **Platform** $\rightarrow$ **Platform** process: the platform ingests data about users and content, runs some logic, and produces an output that shapes the environment agents operate in.

Ranking & Feed Curation The ranking algorithm assigns a score s_c to each piece of content c in the agent's incoming feed and orders it accordingly. The score can be computed from content attributes (topic, recency), engagement signals (likes, shares, replies), source attributes (follower count), and receiver attributes (past engagement history, ideological proximity).

Different combinations of these signals produce qualitatively different dynamics. A chronological feed treats all content equally; an engagement-based feed amplifies popular content and can produce visibility inequality; a similarity-based feed can accelerate the formation of filter bubbles.

Design Recommendation: Treat the ranking algorithm as an *experimental variable* rather than a fixed parameter. It is often the central object of manipulation in TWON experiments.

Content Moderation Content moderation is the set of platform-side rules that determine whether a piece of content remains visible, is demoted, or is removed entirely. Automated moderation can take several forms: hard banning (content is removed), shadowbanning (content remains visible to the author but not to others), and rate limiting (the reach of content or a user is artificially reduced).

In a TWON, moderation operates as a filter on the content pool before ranking is applied. It is worth including when the research question involves the effects of platform intervention on discourse, polarization, or the spread of harmful content.

Algorithmic Amplification Beyond ranking, platforms can actively amplify certain content or users independently of organic engagement signals — promoting content that maximizes watch time, engagement, or ad revenue. This is distinct from other ranking approaches, in that it is not a neutral ordering but an active push toward specific outcomes.

In a TWON, amplification can be modeled as an additive term in the ranking score s_c, or as a separate process that injects content into feeds regardless of the standard ranking logic.

Design Recommendation: Include the notification system only if your research question involves user activation patterns or platform-driven engagement. If activation is modeled independently, this component can be omitted.

1.3 Build the User Model

The user model is the user-side counterpart to the platform model. Where the platform defines the environment — the action space, the information structure, the rules of engagement — the user model defines who acts within that environment and how. Every user model, regardless of its level of granularity, must specify five things: what users *want* (preferences), what limits them (constraints), how they *choose* (decision rule), how they *change* over time (internal state update), and how they *differ* from one another (heterogeneity).

1.3.1 Preferences

Users are reward-driven: they tend to repeat behaviors that were experienced as rewarding in the past. The key modeling decision is therefore not whether agents have preferences, but *what counts as a reward* — and this choice must follow directly from the research question.

The most common reward signal in online social network models is **social feedback**: likes, replies, shares, and other forms of engagement from other users. This is the appropriate default when the research question concerns visibility dynamics, influence, or user engagement. Two additional reward types are worth considering depending on the platform and question. **Financial reward** is relevant for monetized platforms, where content creators earn income proportional to engagement. **Ideological alignment** — the reward an agent derives from consuming or sharing content consistent with its own views — is relevant when the research question concerns polarization, filter bubbles, or selective exposure.

Preferences can also be modulated to encode specific biases or behavioral tendencies. **Confirmation bias** can be implemented by assigning a higher reward weight to ideologically aligned content, causing agents to preferentially engage with it. **Success-driven activity** shifts the reward source from receiving feedback to *influencing* others, producing qualitatively different dynamics in which agents who successfully persuade their neighbors become disproportionately active Horn et al. (in press). These modifications are not features to be added by default; they should be included only if the mechanism they represent is identified as causally relevant to the research question.

Design Recommendation: Specify the reward function before implementing agent behavior. Every behavioral choice in the model — whether to post, react, or log off — will be evaluated against this function.

1.3.2 Constraints

Behavior is always associated with costs. Engaging in online activity — logging in, reading content, posting, reacting — requires time and effort. Agents therefore have a limited **resource budget** that is depleted by action and replenished over time.

Two budget levels are worth distinguishing. A **global budget** captures the total time and energy available to an agent across an entire session or simulation period. A **round budget** caps how much an agent can do within a single activation — even if the global budget is not exhausted, an agent cannot act indefinitely within one round. Both deplete with each action and recover between rounds at a specified rate.

Heterogeneity in budget parameters is a straightforward way to encode real-world differences in platform usage: heavy users have a larger global budget and faster recovery; occasional users have smaller budgets and engage in fewer actions per session.

Design Recommendation: If the research question concerns long-run aggregate dynamics, a coarse budget model suffices. If within-session behavior — such as the sequence of actions an agent takes once logged in — is central to the research question, a finer-grained budget model is needed.

1.3.3 Decision Rule

The decision rule specifies how users translate preferences and constraints into actual behavior on the platform. It is the mechanism by which a reward function becomes an action.

Our suggestion for formal TWON models is the **myopic best-response rule**: at each decision point, an agent selects the action with the highest expected reward based on *past experience*, without anticipating future consequences or the reactions of other agents. In the common case where the user chooses between two options, this translates the reward difference into a selection probability via a logistic function — producing a stochastic decision that is more likely to favor the higher-reward option but not deterministic.

This decision rule incorporates four aspects of bounded rationality that depart from classical rational choice. First, users are **backwards-looking**: they base decisions on past rewards rather than forward predictions. Second, they are **myopic**: they do not anticipate how their actions will affect others or how others will respond. Third, they exhibit **random deviations** from otherwise preferred options, preventing the system from collapsing into deterministic cycles. Fourth, they may **switch between behavioral modes**: for instance, applying decreasing marginal reward when deciding to log on, but shifting to a more compulsive consumption pattern once a session has begun — a form of within-session

addictive behavior.

Design Recommendation: Include only the dimensions of heterogeneity that your observables require. Each added dimension increases the parameter space and complicates calibration. If the research question does not require distinguishing agents by budget, a uniform budget is preferable. If initial opinion distribution does not affect the observable of interest, a homogeneous starting state is simpler and more interpretable.

1.4 Build the Scheduler

The scheduler governs when users act. While the user model specifies what a user wants and how it chooses, and the platform model specifies the environment it acts in, the scheduler determines the temporal structure of the simulation: who is active, when, and in what order. It operates at two distinct levels. At the *macro level*, it governs when users log on and off the platform — transitions between active and inactive states driven by behavioral decisions. At the *micro level*, it governs what happens within a session: the sequence of actions an agent takes, the budget it consumes, and how individual actions are ordered relative to others. Both levels require explicit design choices, and both connect back to components already specified in the agent and platform models.

1.4.1 Discretizing Time

The first and most fundamental scheduling decision is how to represent time. The standard approach is a **round-based** model: time proceeds as a sequence of discrete steps, and at each step all active users act simultaneously on the same frozen system state. This synchronous structure has a consequential implication: a user's action in round t is not visible to other users until round $t + 1$. A post produced by user i during round t will not appear in user j's feed until the following round, meaning users cannot respond to each other within the same time step. This is a deliberate modeling approximation, not merely an implementation convenience, and it is reasonable when rounds are short relative to the timescale of the dynamics of interest. When rounds are long, however, this approximation may suppress interaction patterns that are potentially important.

An alternative is a **Poisson process** model: rather than activating all users at every step, users are activated stochastically at individual rates, so that at any given moment only one or a few users act. This avoids iterating over the entire population at every time step and scales better to large agent populations. It also produces a more realistic distribution of inter-event times, recovering the bursty activity patterns observed in empirical data without the need of extra assumptions. The cost is that the synchronous guarantee is lost: the system state changes continuously, and the causal structure of who

responded to what becomes harder to track. The choice between round-based and Poisson scheduling should be made on the basis of population size, on the fraction of agents performing an action at each time steps, and whether within-round simultaneity is a substantive assumption or merely a computational convenience.

1.4.2 The Activation Decision

Before a user acts within a round, it must decide whether it is online at all. This is not a parameter set by the researcher but a behavioral decision modeled the same way as any other. One possible implementation is to assume that the user compares the expected reward of logging on against the cost of doing so, given past experience. The inputs to this decision include the expected duration of the session, the expected social feedback accumulated while online, and the personal value derived from content consumption. Platform-driven signals feed directly into this calculation: a recent notification increases the expected reward of logging on and introduces a FOMO cost associated with staying offline; this could be implemented as a negative utility that decays exponentially with time since the last notification. Login and logoff are therefore emergent outcomes of the agent's preference structure, not free parameters. The implication for model design is that the activation pattern of the population — how many agents are online at any given round, how long sessions last, how frequently users return — is jointly determined by the agent model and the platform's notification system, and will change if either is modified.

1.4.3 Action Sequencing Within a Session

Once online, a user faces a sequence of decisions: whether to read incoming content, whether and how to react to each item, and whether to produce new content of its own. The order of these decisions must be specified explicitly. A possible choice is feed-first: the user receives its ranked feed, consumes some share of it, reacts to items it finds sufficiently rewarding, and then decides whether to post. The round budget from Section 4 enforces the constraint: each action consumes a fixed amount of budget, and when the round budget is exhausted the agent stops acting for that round, even if its global budget is not yet depleted. If an action — such as producing a long post — consumes more than the remaining round budget, the agent remains committed to it across multiple consecutive rounds, becoming effectively unavailable for other interactions until the action is complete. This can produce patterns of users who are intermittently absent from reactive behavior because they are engaged in a time-costly production task.

Researchers who are not interested in within-session dynamics can simplify this component substantially by assigning each agent a fixed number of actions per round and ignoring the budget mech-

anism. This is a legitimate simplification when the research question concerns aggregate or long-run dynamics rather than the fine-grained sequence of individual interactions.

1.4.4 Stopping Rules

A TWON has no natural equilibrium. Unlike many formal models of social dynamics, the system does not in general converge to a fixed point at which further rounds produce no change — user states, content pools, and network structure continue to evolve as long as the simulation runs. The simulation must therefore be stopped by an external rule, typically a fixed number of rounds specified before the simulation begins. This choice is non-trivial. Too few rounds and the measurements will reflect transient initialization effects rather than the dynamics of interest. Too many rounds and the system may drift into regimes that are artifacts of model assumptions rather than the mechanisms under study. Two approaches to calibration are available. The first is **empirical calibration**: fitting parameters to observed behavioral data such as session length distributions, posting frequencies, or inter-arrival times between logins. Taken to its limit, this approach yields a fully learned scheduler. The second is **sensitivity analysis**: running the simulation across a range of parameter values and verifying that the qualitative results do not depend on the specific choice. When empirical data is unavailable, sensitivity analysis is the minimum due diligence required before reporting results. Parameters to which the target observable is highly sensitive should be flagged explicitly as sources of uncertainty.

2 Four Purposes - Four TWONs

In Section 1, we outlined the core structure of a TWON based on the model structure put forward in Deliverable 2.1 and emphasized that this general architecture must be instantiated for a specific online social network as well as a specific research question. In the following, we present four TWONs, each developed to address a distinct research question and applied to a particular empirical context.

The four TWONs differ along three core dimensions. Together, these dimensions capture key trade-offs that modelers must consider when developing a TWON:

- **Realism:** TWONs vary in the extent to which they realistically represent users, content, and platform dynamics. Three of the TWONs we developed, for example, use Large Language Models (LLMs) to represent the content that users create and share. Moreover, these LLMs are fine-tuned with empirical data from OSNs to mimic the content produced by real users. Another TWON is much less realistic in this respect, representing content only as a numerical value that encodes the position a user holds on a specific issue.

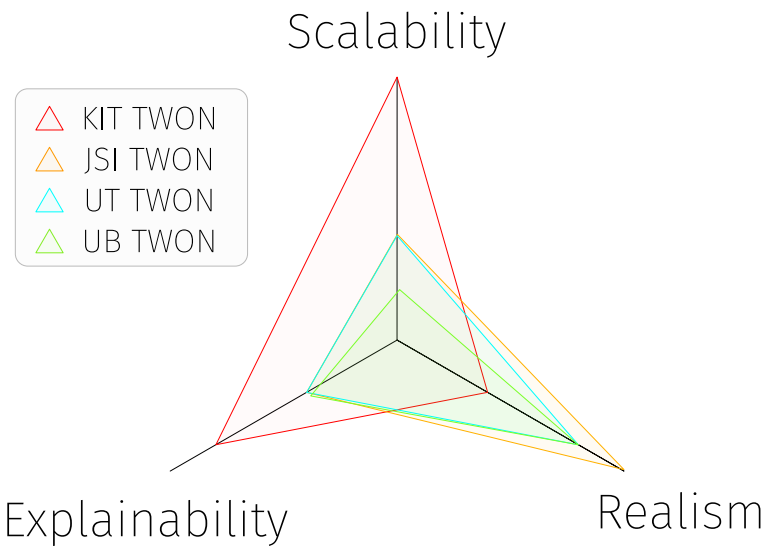


Figure 2: Classification of the LSS models of TWON according to three dimensions, scalability, explainability and realism.

- **Scalability:** Highly realistic TWONs tend to be computationally demanding. As a consequence, they can typically be used only to study relatively short time spans (e.g., a few days), communities with relatively few users (e.g., around 100), and a limited number of experimental treatments and Monte Carlo replications. Less realistic, more abstract models, by contrast, are far less computationally constrained and can be scaled to larger populations, longer periods, and richer experimental designs.

- **Explainability:** TWONs also differ in how easily their predictions can be interpreted and explained. Models often produce unexpected or counterintuitive outcomes, prompting the question of why they generate the observed dynamics. Answering this question can be extremely challenging when models generate compley dynamics. One important source of complexity is the user model. LLM-based user models, for instance, are black boxes that are difficult to interpret, and tailoring them to real users only increases this opacity. What is more, LLMs genrate complex text that also needs to be interpreted. This complexity makes it extremely hard to explain why LLM-based TWONs generate the predictions they imply. Another source of complexity is the interaction between users, which is governed by the platform model. When complex ranking algorithms are implemented, understanding the model's predictions can become very difficult to explain. In general, abstract models tend to be more explainable, whereas highly realistic models are harder to interpret.

Figure 2 positions the four TWONs in terms of their realism, explainability, and scalability on an ordinal scale. It shows that the KIT-model scores high on scalability and explainability but does so on the expense of realism. The TWONS developed at UT and UB used sophisticated LLMs to realistically represent user behavior and, accordingly, score very high on the realism scale. The most realistic TWON was developed at JSI, since here also important parts of the platform model were calibrated from empirical data.

3 KIT TWON

3.1 Purpose of the KIT TWON

The KIT TWON was developed to address a fundamental methodological gap in computational studies of online polarization. Existing opinion dynamics models conflate two distinct mechanisms: (1) how individuals update their opinions when exposed to new information, and (2) which information individuals are exposed to in the first place. This conflation makes it impossible to isolate the specific contribution of algorithmic curation to collective opinion dynamics.

As Figure 3.1 shows, the KIT-TWON sacrifices user realism in favour of computational tractability and experimental control. Rather than modelling users with large language models or empirically trained behavioral profiles, users follow simple, well-understood update rules from the opinion dynamics literature. This design choice enables three things that would otherwise be infeasible:

- running hundreds of replicas per parameter combination to obtain statistically reliable estimates,
- sweeping broadly across the joint space of user psychology and ranking algorithm parameters, and
- cleanly attributing observed outcomes to either the update mechanism or the selection mechanism.

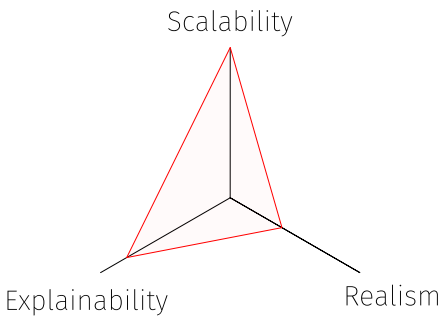


Figure 3: Classification of the KIT-TWON according to three dimensions, scalability, explainability and realism.

The central hypothesis driving the design is that ranking algorithms can override individual-level psychological factors in determining whether a population polarises. Testing this hypothesis requires precisely the kind of controlled, large-scale parameter exploration that the simple-user approach enables.

3.2 Parameters and model setup

3.2.1 Platform model

Network structure The social network is modelled as an Erdős–Rényi random graph with $n = 1000$ agents and edge probability $p = 0.02$, yielding an expected degree of approximately 20 neighbors per agent. The network is static: edges do not change during a simulation run. Each agent can only see posts authored by their direct network neighbors.

Content storage Each agent maintains a circular buffer of $H = 50$ posts, representing their recent posting history. Each post stores its the position on the debated issue held by the post's creator at the time of creation and a cumulative like count. The buffer is updated once per timestep: the agent's newest post overwrites the oldest slot.

Ranking of incoming messages At each timestep, every agent is shown $k = 1$ posts selected from the set of unseen posts authored by their neighbors. The selection is governed by one of six ranking algorithms:

- **Random:** uniform random selection from all unseen neighbour posts (baseline).
- **Engagement:** posts are sampled with probability proportional to their cumulative like count raised to a power $\alpha \in \{0.1, 0.5, 1.0, 2.0, 5.0\}$. Higher α amplifies the rich-get-richer effect.
- **Closest:** posts are ranked by opinion similarity to the viewing agent, prioritising ideologically proximate content (filter bubble).
- **Narrative:** posts are ranked by proximity to a fixed target opinion $\tau \in \{0, 0.25, 0.5\}$, simulating a platform with an editorial or algorithmic agenda.
- **Diverse_Engagement:** combines engagement weighting with a diversity correction that penalises opinion similarity.
- **User_Success:** posts are sampled with probability proportional to the author's cumulative total likes raised to $\alpha \in \{0.1, 0.5, 1.0, 2.0, 5.0\}$, creating agent-level rather than post-level inequality.

3.2.2 User model

User activation All users are activated synchronously at every timestep. There is no dormancy or variable activity rate.

User actions At each timestep a user: (1) receives one post selected by the ranking algorithm, (2) reads it, (3) likes it if and only if the post opinion falls within their confidence bound ε, and (4) updates their own opinion accordingly. Liking is thus not a purely social signal but a direct expression of opinion proximity, and like counts feed back into engagement-based ranking in subsequent steps.

Content creation After reading, every user publishes one new post per timestep. Post content is simply the user's current opinion value — there is no natural language, no rich media, and no strategic communication. This is the principal simplification of the KIT TWON relative to LLM-based alternatives.

The opinion update follows the Bounded Confidence Model (BCM). Let $o_i(t)$ denote agent i's opinion at time t and o_j the opinion expressed in the post shown to i. The update rule is:

$$o_i(t+1) = o_i(t) + \mu \cdot (o_j - o_i(t)) \cdot \mathbf{1}\left[|o_j - o_i(t)| < \varepsilon\right] \tag{1}$$

where $\varepsilon \in [0, 0.4]$ is the confidence bound and $\mu = 0.1$ is the convergence rate. Low ε represents closed-minded agents; high ε represents open-minded agents. Opinions are initialized uniformly at random in $[0, 1]$.

3.2.3 Technical Infrastructure

The simulation is implemented in Python using NumPy for vectorised array operations and igraph for network construction. Each simulation run is self-contained and stateless between runs, enabling embarrassingly parallel execution. Experiments were run on the HoreKa HPC cluster at KIT using SLURM job arrays, with one parameter combination per job. Reproducibility is ensured through Singularity containers. Results are stored in compressed NPZ format with JSON configuration sidecar files.

3.3 Example finding

The central search question to be answered with this TWON addresses the effects of ranking algorithms on opinion polarization. To answer it, we ran simulations with the six described ranking algorithms and ten different values of paramter ε. For each of the 60 parameter combinations, we conducted 100 independent simulation runs and always measured polarization when runs had reached a state of equilibrium. Polarization was measured as $4\times$ variance of the opinion distribution Figure 4 shows the findings.

The key finding is that ranking algorithms determine whether polarization persists at high ε, overriding individual-level psychology. Under most rankers — including the Random baseline — polarization decays to zero once agents are sufficiently open-minded ($\varepsilon \gtrsim 0.25$), confirming the classical BCM result.

Two configurations violate this pattern, however:

- **Closest** (filter bubble): polarization remains near 0.33 across the *entire* ε range. Even maximally open-minded agents remain polarized when the algorithm systematically shields them from distant opinions.
- **Narrative,** $\tau = 0$ (extreme agenda): polarization *increases* with ε because the algorithm creates a two-cluster structure. Users whose opinion falls within the confidence bound of the target get absorbed into a cohesive group near $o = 0$, while users whose opinion is too distant from the target never accept the content and form an isolated residual cluster. As ε widens, the absorption band grows and more users are pulled in, but the gap between the absorbed cluster and the left-behind group increases — driving polarization up rather than down.

These results demonstrate that algorithmic selection can both sustain and amplify polarization independently of how psychologically open users are — a finding that would be invisible in models that do not separate the two mechanisms.

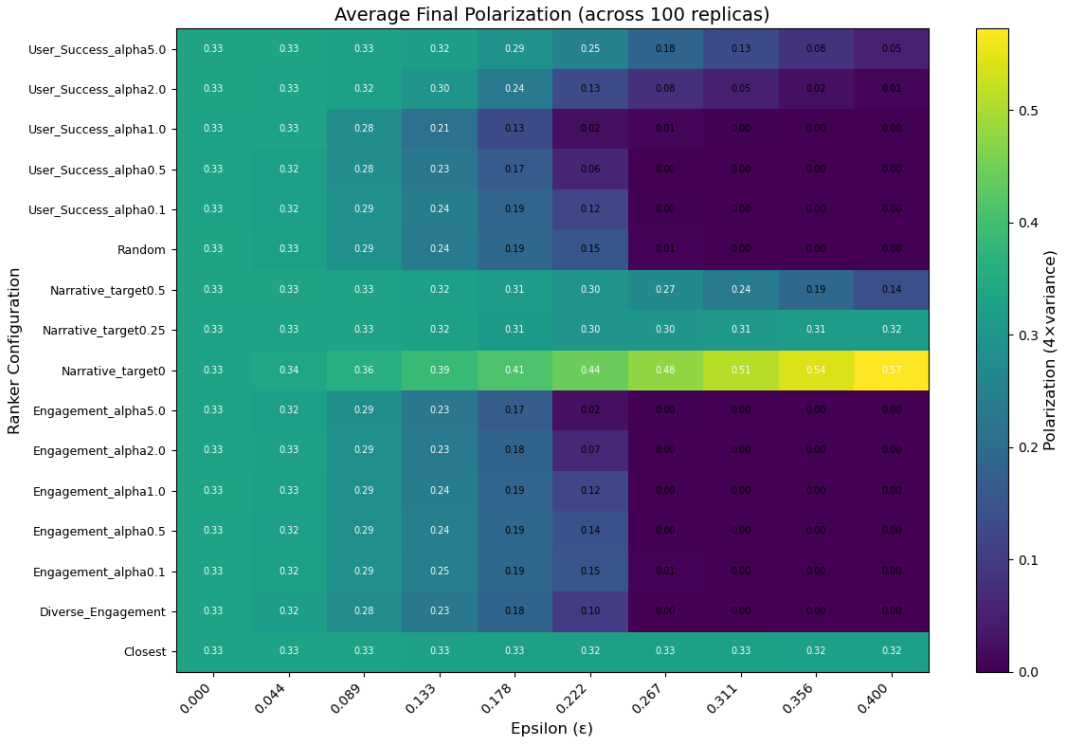


Figure 4: Average final polarization ($4\times$variance) across 100 replicas as a function of the confidence bound ε (x-axis) and ranking algorithm (y-axis). Most rankers produce zero polarization at high ε. The Closest ranker sustains polarization throughout, and the Narrative ranker with extreme target ($\tau = 0$) increases polarization as agents become more open-minded.

4 JSI TWON

4.1 Purpose of the TWON

The JSI TWON was developed to bridge the methodological gap between computational social science and generative modeling by creating a data-driven multi-agent simulation framework grounded in real-world behavioral data. Unlike traditional agent-based models (ABM) that rely on simplified rules or fixed opinion vectors, this TWON utilizes Large Language Models (LLMs) to reproduce the linguistic and cognitive richness of real online discourse. The primary aim is to provide a validated, controlled testbed to study how algorithmic curation and network structure influence collective social phenomena, such as filter bubbles and echo chambers. By integrating temporal behavior prediction with persona-based content generation, the model investigates whether synthetic discussions can realistically replicate the linguistic, temporal, and structural properties observed on platforms like Reddit. To this end, we tailored the TWON to the platform Reddit, implementing network structures and user activity patterns observed on Reddit.

As Figure 4.1 visualizes, the JSI TWON scores very high in terms of realism, since both the user model and the platform model have been calibrated with data from a real online social network (Reddit). Obviously, this comes at the expense of scalability and explainability.

4.2 Parameters and model setup

4.2.1 Platform model

Network structure The social structure is derived from real Reddit discussion networks across three domains: Technology (33 communities), Climate (14 communities), and COVID-19 (7 communities). Directed interaction networks are inferred from historical user-to-user commenting patterns, which are treated as indicators of attention or following behavior.

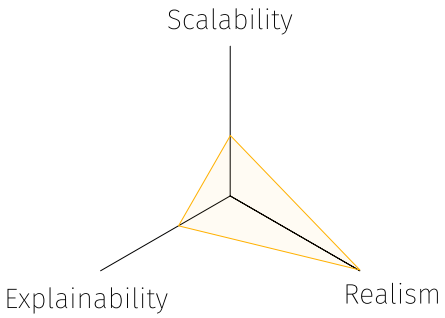


Figure 5: Classification of the JSI TWON according to three dimensions, scalability, explainability and realism.

Ranking of incoming messages The framework implements configurable exposure ranking mechanisms to study their impact on social growth. Users are exposed to content through six variations of ranking strategies:

- Global/Local Chronological: Preserves interaction diversity by displaying the most recent content.
- Global/Local Engagement: Prioritizes globally popular or locally high-activity threads, often amplifying influence inequality.
- Global/Local Hybrid: Combines engagement signals with social proximity, potentially accelerating the formation of echo chambers.

4.2.2 User model

User activation Instead of fixed schedules, user activation is governed by a learned scheduler implemented via a Temporal Fusion Transformer (TFT). This scheduler is trained on real Reddit data to predict the next user action—including who acts, when they act, and what topic they address—reproducing realistic burst patterns and conversation lifecycles

User actions Agents perform two primary actions: posting and commenting. These actions are conditioned by agent profiles that encode long-term behavioral and stylistic attributes, such as dominant emotional tone and topical preferences, ensuring behavioral consistency over time.

Content creation Content is generated using LLaMA-2-7B-chat models specialized through instruction-based LoRA (Low-Rank Adaptation) fine-tuning for each domain (Technology, Climate, COVID). To ensure factual and conversational grounding, the model employs Retrieval-Augmented Generation (RAG), using FAISS to select semantically relevant historical posts and comments as context for the LLM.

4.2.3 Technical Infrastructure

The system is implemented as a modular pipeline using Python. It utilizes MongoDB for storing agent states and content, FAISS for dense vector embedding retrieval, and the Hugging Face framework for managing the LLaMA-2 models. LoRA is used for parameter-efficient fine-tuning to reduce computational costs while maintaining domain-specific fluency.

4.3 Example finding

The key finding is that ranking algorithms substantially shape emergent social structures, regardless of the underlying agent behavior. While chronological exposure maintains a diverse spread of interactions, engagement-based ranking amplifies dominant threads and increases influence concentration. In large-scale simulations within the Technology domain, hybrid ranking (using social proximity) was found to reduce cross-cluster interactions and increase within-cluster density. This demonstrates that exposure mechanisms and network evolution can lead to modular segregation and echo chamber consolidation even when the initial temporal modeling is grounded in real data.

5 UT TWON

5.1 Purpose of the TWON

The UT TWON was developed to reconcile the need for experimental flexibility with the demand for empirical realism in modeling online ecosystems. To achieve this, it employs a dual-layer architecture. The foundational layer consists of an abstract, modular simulation framework designed to support systematic variation across core environmental and behavioral components, including ranking algorithms, network topologies, and agent cognitive models. This structural decoupling enables researchers to isolate, interchange, and evaluate specific mechanisms without redesigning the underlying simulation infrastructure.

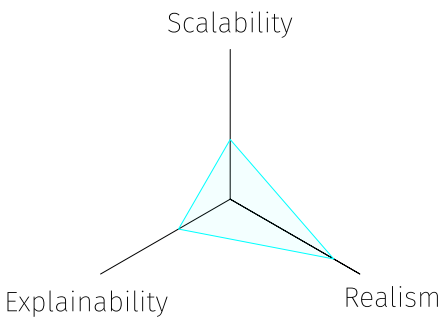


Figure 6: Classification of the UT TWON according to three dimensions, scalability, explainability and realism.

Built upon this flexible foundation is the second abstraction layer: a fully data-driven implementation designated as the WP3 simulation. This instantiation maximizes reliance on LLMs to evaluate their operational robustness and out-of-distribution prediction capabilities across varying agent configurations. To achieve this, the simulation derives its operational parameters directly from empirical X data. Rather than employing generalized user archetypes, WP3 replicates specific, real-world users for whom supplementary validation data exists. This replication is accomplished by initializing LLM agents with context windows seeded by the users' actual historical posts and derived biographies **?**. By grounding agent personas in authentic histories, the simulation provides a controlled, high-fidelity environment to test whether LLM-driven agents can accurately

predict unseen behavioral trajectories.

5.2 Parameters and model setup

The WP3 simulation is fundamentally modular, relying on a configuration-driven architecture (via YAML files and CLI arguments) that allows for the dynamic instantiation of network topologies, ranking algorithms, and empirical agent parameters without structural code alterations.

5.2.1 Platform model

Network structure The network topology is fully configurable via the underlying NetworkX integration. The WP3 simulation dynamically instantiates either a complete graph or a Barabási–Albert scale-free network to model preferential attachment dynamics. We tested the range of 1000 to 4000 agents.

Content storage Content is managed via a global `Feed` object that aggregates agent posts. To ensure ecological validity from timestep zero, the feed is initialized with authentic historical data (e.g., the last two empirical posts from each user's dataset). For context processing, each agent maintains a bounded, parameterized sliding memory window of recent interactions (configurable via `agent_memory_length`, yielding an effective memory capacity of $2 \times length$ actions). This dynamically constrains the LLM context size while preserving relevant short-term interaction history.

Ranking of incoming messages Ranking is handled by a modular ranking interface. The environment supports three primary selection algorithms, instantiated at runtime and governed by a configurable `persistence` (how long after posting can posts be accessed) parameter:

- **RandomRanker:** Uniform random selection from the feed, serving as an uncurated baseline.
- **ChronologicalRanker:** Temporal ordering, prioritizing the most recently generated content.
- **SemanticSimilarityRanker:** Content is ranked by semantic proximity using text embeddings (via local or API-based embedding models), simulating recommendation systems optimized for homophily and ideological resonance.

5.2.2 User model

User activation Agent activation is stochastic and empirically grounded rather than fixed. When configured (via `use_auth_activation`), an agent's base probability for activation, reading volume, and

posting frequency are algorithmically derived from their empirical dataset counterpart (utilizing their actual historical `posts_per_day` metric).

User actions Upon activation, users process content curated by the active platform ranker and execute actions (reading, evaluating, or posting). These actions are mediated by the user's cognitive state and empirical profile. Through modular toggles (`use_auth_profile`, `use_auth_history`, `dynamic_cognition`), users can be configured to utilize their authentic bios, inferred cognitive frameworks, and historical post data, or update their belief systems dynamically to maintain longitudinal behavioral consistency.

Content creation Content generation is driven by Large Language Models (accessed via the `WP3LLM` API wrapper targeting models such as Llama-3.1-8B-Instruct). The LLM is prompted using specific agent instructions and optionally grounded in the user's authentic conversational history and configured cognitive stance. The framework also supports the injection of static, high-influence institutional personas (e.g., public health authorities like the RKI in a COVID-19 scenario) to evaluate asymmetric information propagation.

5.2.3 Technical Infrastructure

The foundational TWON-LSS framework is implemented in Python, utilizing an object-oriented architecture defined by abstract base classes to ensure modularity. The WP3 simulation operationalizes this architecture, employing a highly concurrent execution model to optimize computational throughput. Specifically, the system leverages multiprocessing to parallelize computationally intensive operations, such as semantic content ranking, alongside multithreading during the agent execution stage, while awaiting asynchronous LLM API responses.

The generative backend is powered by a dynamic vLLM deployment hosted on RunPod, enabling elastic scaling of language model instances in response to real-time simulation demand. For standard workloads, typically executing four simultaneous simulation pipelines, the infrastructure utilizes a distributed cluster of 3 to 5 NVIDIA RTX 4090 GPUs. State persistence and experimental reproducibility are maintained by serializing output data (network graphs, agent states, and feed histories) into structured JSON formats, managed by YAML-based configuration tracking.

5.3 Example finding

Table 1 presents the similarity between synthetic and authentic feeds across three agent configurations (Naive Prompting, Data-Driven, and Cognition Update) evaluated under in-distribution (Snap-

Method	Snapshot (ID)	Snapshot Covid (OOD)
Nela		
Naive Prompting	0.761	0.815
Data-Driven (Static)	**0.828**	**0.848**
Cognition Update (Dynamic)	0.820	0.811
Spacy		
Naive Prompting	0.872	0.894
Data-Driven (Static)	**0.940**	**0.959**
Cognition Update (Dynamic)	0.938	0.958
Tweet Eval		
Naive Prompting	0.841	0.869
Data-Driven (Static)	**0.940**	**0.946**
Cognition Update (Dynamic)	0.933	0.942

Table 1: Similarity between synthetic and authentic feeds across three methods, evaluated on in-distribution (Snapshot) and out-of-distribution (COVID) data. Compare method to **?**

shot) and out-of-distribution (Snapshot Covid) conditions. System performance is quantified using linguistic (Nela, Spacy) and semantic (TweetEval) metrics. Three primary observations emerge from the evaluation:

Superiority of Empirically Grounded Agents The Data-Driven configuration consistently outperforms Naive Prompting across all metrics and environments. Seeding the LLM context windows with authentic user histories and inferred behavioral profiles significantly improves the empirical realism of the generated discourse.

Robustness to Out-of-Distribution Data The predictive fidelity of the Data-Driven model remains highly stable when applied to out-of-distribution (COVID-19) scenarios. This demonstrates the capacity of appropriately initialized LLM backends to generalize user behavioral trajectories to unseen topical contexts without structural degradation.

Diminishing Returns of Dynamic Cognition The Cognition Update mechanism fails to yield performance gains over the static Data-Driven baseline, often resulting in marginally lower similarity scores. This indicates potential ”over-alignment”, where the continuous updates of an agent’s cognitive state through the LLM itself introduces behavioral drift rather than improving predictive accuracy.

6 UB TWON

6.1 Purpose of the TWON

This TWON was developed to address two interconnected challenges in LLM-based social simulation: (1) understanding how algorithmic personalization drives polarization in social media ecosystems, and (2) validating whether LLM-powered users can serve as reliable proxies for human social media users. It combines a fully operational simulation framework (RecSysLLMsP) with large-scale empirical benchmarking of agent accuracy.

The framework serves a dual purpose. First, the Recommender Systems LLMs Playground (RecSysLLMsP) provides a controlled environment to observe how progressive algorithmic personalization transforms engagement patterns, network structure, and affective polarization across simulated social media interactions. Second, a complementary validation study benchmarks the behavioral fidelity of LLM-based agents using over 120,000 unique agent-persona combinations derived from 1,511 human

participants, establishing empirical bounds on when and how LLM agents can predict human social media reactions.

The central hypothesis is that recommender systems can amplify polarization independently of user psychology, and that LLM agents grounded in psychometric and demographic data can reproduce this process with measurable fidelity. Testing whether the two supports this hypothesis requires both the simulation infrastructure to manipulate algorithmic parameters and the validation methodology to assess user model realism.

6.2 Parameters and model setup

6.2.1 Platform model

Network structure The social network is modeled as a dynamically evolving directed graph $G = (V, E_t)$, where vertices V represent 100 agent prompts and edges E_t correspond to "follow" relations at time t. Unlike static network models, the topology evolves endogenously through agent interactions: follow actions create new edges but edges cannot be removed within a simulation cycle. Initial network density is determined by user connectivity parameters drawn from empirical survey distributions, approximating scale-free properties observed in real social media networks.

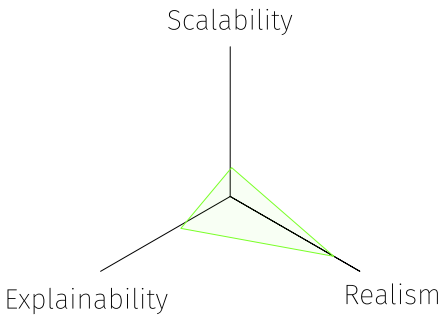


Figure 7: Classification of the UB TWON according to three dimensions, scalability, explainability and realism.

Content storage Each user maintains a history of generated posts. The global content pool P_t contains all posts available at timestep t, categorized into five types: Entertainment posts (E), News-inspired posts (N), Primed posts on global issues (P), and Agent-generated posts from followed (AF) or unfollowed (AU) agents. Each post carries sentiment annotations ($s_p \in [1, 5]$) and stance labels (Pro-X, Neutral, Contra-Y).

Ranking of incoming messages The recommender engine $R(t)$ implements a five-stage personalization trajectory:

- **Step 1 (Universal):** Random exposure from complete post corpus ($\gamma = 0$). Baseline condition with no personalization.

- **Step 2 (25% Personalized):** Quarter of feed adapted using Posts Personalizer based on agent characteristics.
- **Step 3 (50% Personalized):** Half of content individualized using accumulated interaction history.
- **Step 4 (75% Personalized):** Majority of exposure determined by similar-agent affinity vectors.
- **Step 5 (Full Personalization):** Complete individualization ($\gamma \to \infty$). All exposure conditioned on agent profile D_i.

The exposure probability is governed by:

$$P_{i,j,t} = \frac{\text{sim}(a_i, a_j)^\gamma}{\sum_k \text{sim}(a_i, a_k)^\gamma}$$

where sim() computes agent similarity across personality and attitude dimensions, and $\gamma_t = 0.25(t-1)$ increases linearly across steps.

6.2.2 User model

User activation All 100 users are activated synchronously at every timestep. Each user is defined as $a_i = \{D_i, F_i(t), R_i(t), C_i(t)\}$, where D_i denotes fixed psychometric dimensions, $F_i(t)$ represents followed agents, $R_i(t)$ denotes reactions, and $C_i(t)$ represents generated content.

User actions At each timestep, users: (1) receive posts selected by the ranking algorithm, (2) react with like/dislike based on ideological and emotional alignment, and (3) potentially follow the post author. The probability of liking post p by agent i is modeled as:

$$P(\text{Like}_{i,p}) = \sigma(\alpha \cdot (1 - |PA_i - \text{Stance}_p|) + \beta \cdot (7 - |CS_i - s_p|))$$

where $\alpha = 0.45$ and $\beta = 0.55$ calibrate ideological and emotional sensitivity, and σ is the logistic function. Agents are grounded in empirical psychometric and demographic data from Serbian social media users ($N = 1{,}511$ for validation). Dimensions include: Big Five personality traits (Openness, Conscientiousness, Extraversion, Agreeableness, Neuroticism), political attitude (7-point liberal-conservative scale), cognitive style (analytical vs. emotional processing), social connectivity, and engagement level.

Content creation Each agent generates one new post per timestep, with emotional tone modulated by the aggregated sentiment of previously liked posts. This implements emotional contagion: individ-

ual sentiments become structural variables affecting network-level dynamics. For the validation study, 27 LLMs were tested as generative backends, including models from OpenAI, Google, xAI, Mistral, Meta, Anthropic, DeepSeek, and others.

6.2.3 Technical Infrastructure

The RecSysLLMsP simulation is implemented in Python with NetworkX for graph operations. The validation pipeline queries 27 LLMs via API under identical prompt structures, generating 81 agent versions per participant (3 persona specificity levels $\times$ 27 LLMs). Persona descriptions are constructed at three levels: demographics only, attitudes/values only, and full combined profiles. Monte Carlo replications (10 runs per configuration) ensure statistical reliability. Data and code are available via the Open Science Framework (OSF).

6.3 Example finding

Simulation Results (RecSysLLMsP): Network modularity Q_t and affective polarization AP_t both exhibited clear upward trajectories across personalization stages. Mean Q increased from 0.22 at Step 1 to 0.68 at Step 5. Community detection algorithms consistently identified two to four dominant clusters by Step 5, corresponding to ideological divisions on primed topics (climate change, Gaza, Ukraine). Engagement intensity followed an inverted U-shape: moderate personalization (Steps 2–3) maximized attention and emotional investment ($EI = 2.11$), while full personalization reduced variety and led to saturation ($EI = 1.35$). This demonstrates that algorithmic curation can both sustain and amplify polarization independently of user psychology.

Validation Results (Agent Accuracy Benchmark): LLM agents achieved 70.7% accuracy in predicting human social media reactions across 120,000+ agent-persona combinations, exceeding the 50% baseline. The choice of LLM was the strongest predictor of performance, with top models (Grok-3-mini-fast, LLaMA-3.3-70B, Gemini-2.5-Flash, GPT-5.2) outperforming lowest-ranked models by approximately 13 percentage points. Richer persona descriptions produced small but consistent accuracy gains ($\sim$2 percentage points). Entertainment/lifestyle posts were predicted more accurately than news/politics posts ($\sim$4 percentage points difference). A consistent positivity bias was observed: agents predicted reactions to positive content up to 11 percentage points more accurately than reactions to negative content.

References

Sophia Horn, Sven Banisch, Veronika Batzdorfer, Andreas Reitenbach, Fabio Sartori, Daniel Schwabe, and Michael Mäs. Success-driven user activity contributes to online polarization. *JASSS-Journal of Artificial Societies and Social Simulation*, in press. 8, 10, 13

Marijn A Keijzer and Michael Mäs. The complex link between filter bubbles and opinion polarization. *Data Science*, 5(2):139–166, 2022. 8

Douglass C. North. *Institutions, Institutional Change and Economic Performance*. Cambridge University Press, Cambridge, 1990. 10

Eli Pariser. *The filter bubble: What the Internet is hiding from you*. Penguin UK, 2011. 8

twon

twin of online social networks

Contact us

Damian Trilling

Project Coordinator

+31 62 782 7904

d.c.trilling@uva.nl

University of Amsterdam
Postbus 15791
1001 NG Amsterdam

Funded by
the European Union