Ŧrust: Learning where patterns come from and how they change through time

John Ash

Independent researcher

12 November 2026

Abstract

Generative AI can reproduce an idea, an image style, or a musical pattern without telling the user whose contributions shaped it or how those contributions developed. Ŧrust addresses this problem within the model: it adds learned representations of source and time alongside content, so ordinary transformer attention can learn their relationships. This makes it possible to study both the evolution of patterns across contributors and their use in generation. The mechanism has been implemented and demonstrated numerically. It learns relationships across source, time, and content; generates paths from real market data; and supplies representations to a policy that makes effective decisions across varied market conditions. This paper defines the shared architecture and explains its relevance to provenance, credit, and Iris, a model intended to learn from the development of collective knowledge over time.

1 Introduction

Suppose an AI-generated explanation draws on an early hypothesis, later work that tests it, and a widely circulated summary. Those contributions have different roles. The first introduces an idea, the second changes what is known about it, and the third helps it spread. An account of where the explanation came from should be able to distinguish them.

The same problem arises in generated media. An image can combine a visual technique, a subject, and a compositional idea developed by different people at different times. Recognizing what the image resembles does not establish which contributions the generator used. Understanding its origin requires a connection between the history of those contributions and the process that produced the image.

Across research, language, art, and code, patterns are continually introduced, repeated, modified, and combined. This shared, evolving body of patterns is what we mean here by a **pattern commons**. Its history includes the people who originate a pattern and those who refine, connect, or communicate it. Source identifies these contributors. Time distinguishes their contributions within that history. Content represents what each contributed.

Ŧrust makes all three available to the model's existing learning machinery. Transformer attention already learns which information to combine for a task. Adding source and time lets it learn relationships that would otherwise be lost when contributions are treated as anonymous content. The extension supports two connected questions: **how did this pattern develop, and which contributions are being used now?**

2 Related work

Ŧrust combines established learning mechanisms. The transformer supplies the underlying attention operation [1]. Earlier work learns user embeddings alongside content representations for sarcasm detection [2] and speaker embeddings for dialogue generation [3]. These studies establish a practical basis for representing who produced an observation within the model itself.

Learned temporal representations provide a corresponding basis for representing when an observation occurred. Time2Vec develops a vector representation of time that can be incorporated into existing architectures

[4]. Temporal Attention for Language Models incorporates the time associated with texts into self-attention and evaluates the resulting representations on semantic change [5].

The focus of this paper is the joint use of source, time, and content, including an explicit source–time interaction, queries in three directions, and a source-indexed readout of attention. The numerical implementations demonstrate how that combination supports learning relationships, generating trajectories, and making decisions. The component designs are developed separately in *Source Embeddings* and *Timestamp Embeddings* [6, 7].

3 Architecture

3.1 Source, time, and content representations

An observation consists of content, its time, and its source. We write this as $o_i = (c_i, t_i, s_i)$, where i identifies an observation. Content may be a numerical measurement or, in a language or media application, a representation of a passage, image element, or audio segment.

An embedding converts each component into a learned vector that the model can operate on. The same source uses the same source-embedding lookup across its contributions, allowing the model to connect observations from that source over time. A timestamp encoder maps actual time into a representation through which temporal relationships can be learned. It combines learned trend and periodic components, allowing the model to learn both gradual change and recurring temporal relationships. These foundations are described in *Source Embeddings* [6] and *Timestamp Embeddings* [7].

Let $v_i = E_c(c_i)$, $u_i = E_t(t_i)$, and $e_i = E_s(s_i)$ be the content, time, and source representations. With time and source represented in compatible vector spaces, the shared numerical architecture combines them as

$$x_i = [v_i; u_i; e_i; u_i \odot e_i]. \tag{1}$$

The brackets mean concatenation; $\odot$ means elementwise multiplication. The final term explicitly combines source and time. It gives the model a way to represent the same source differently in different temporal contexts while retaining its identity. All of these components participate in the subsequent attention computation.

This construction is the common core of the numerical proof, Market Transformer, and embedded policy. It supplies ordinary attention with information about the contributors and temporal development of the patterns it is learning. The numerical implementations develop these representations through contextual attention and further learned computation; a task-specific output mapping turns the resulting representation into an answer.

3.2 Attention and contextual trust

Attention compares a request with the available observations and combines information from them. In transformer terminology, the request is represented by a **query**; each observation supplies a **key** used for comparison and a **value** carrying information to combine. These are learned projections, so training changes both what counts as relevant and what information is passed onward.

For one attention operation, let q be the query vector and k_i, w_i the key and value for observation i. Over the eligible observations $\mathcal{J}$, ordinary scaled dot-product attention is

$$\alpha_i(q) = \frac{\exp(q^\top k_i/\sqrt{d})}{\sum_{j\in\mathcal{J}} \exp(q^\top k_j/\sqrt{d})}, \qquad z_q = \sum_{i\in\mathcal{J}} \alpha_i(q) w_i. \tag{2}$$

Here d is the key size used in the standard normalization. The weights α_i sum to one; z_q is the information combined by this operation. Eligibility respects the task's information boundary, such as excluding future

observations from a forecast. Attention can first operate among observations to develop their contextual representations, and then read from those representations to answer a request.

Ŧrust uses this familiar operation. The change is that the representations being compared include content, source, time, and their interaction. A useful relationship can therefore depend on a source's behavior during a particular period, on another source's observations, or on how a pattern has changed. A numerical output, a generated sequence, or a policy decision gives the learning process an objective against which to develop those relationships.

This is also the meaning of contextual trust in the architecture: how the model allocates attention for the task at hand. The useful source can change with the question, the content, and the time. Several sources can contribute jointly.

3.3 Query directions

Once the model represents the relationship among content, source, and time, each can become the unknown in a question. Here, context means the source-labelled, time-stamped observations supplied to answer a request. Let $\mathcal{C}$ denote that context. Across the numerical research, related implementations have demonstrated queries in each of the following directions:

Table 1: Query directions across the related numerical implementations. Context consists of source-labelled, time-stamped observations.

Known information	Requested answer	Mathematical form
Source and time	What does the source express then?	$\hat{c} = f_\theta(s, t \mid \mathcal{C})$
Source and content	When does the source express it?	$\hat{t} = g_\theta(s, c \mid \mathcal{C})$
Time and content	Which source corresponds to it?	$\hat{s} = h_\theta(t, c \mid \mathcal{C})$

The parameter symbol θ denotes learned model parameters. These functions name the query operations; an application's output mapping turns the learned answer representation into content, a time, or a source identity.

A question may require retrieving a supplied observation or inferring an answer from a learned relationship. This is why the architecture is more than a store of labelled records. It can learn what relates the records, complete missing information, and generate from those relationships.

4 Provenance within attention

Because the observations retain source and time, the attention used in an operation can be read in those terms. To find the attention allocated to source s during a period B, group the weights belonging to that source and period:

$$T_q(s, B) = \sum_{i \in \mathcal{J}:\ s_i = s,\ t_i \in B} \alpha_i(q). \tag{3}$$

Taking B to cover all represented times gives the source's total attention in that operation. Restricting B preserves the temporal distinction: which part of the source's history was consulted? Content remains identifiable through the individual observations included in the sum.

The actual information combined can be grouped in the same way. For a source across all represented times,

$$z_{q,s} = \sum_{i \in \mathcal{J}:\ s_i = s} \alpha_i(q) w_i, \qquad z_q = \sum_s z_{q,s}. \tag{4}$$

Thus the attention distribution describes how the operation allocates its read, while the grouped value vectors describe what it combines. These are quantities in the computation that produces the answer. An inverse source query, by comparison, infers a source from observed content; that is a different operation from recording which contributions generation actually used.

This distinction matters for media origin. A generator's source-aware computation can be inspected rather than asking a second model to guess an origin from resemblance. Where intermediate states already combine contributions, their earlier source relationships must also be followed to describe the full provenance of the output. The readout above defines the contribution at an attention operation; tracing those relationships through generation is the basis of the broader media-provenance proposal.

5 Numerical demonstrations

The numerical setting makes it possible to examine these relationships directly. Sources provide values at known times, so the relevant source, requested time, and correct answer can be checked against the generated output.

5.1 Source selection and inverse queries

In the original trust-selection proof, which source is reliable changes over time. The reliable source supplies the true value; the other sources supply noise drawn from the same distribution. The value itself does not reveal which source should be trusted. Training compares the model's numerical answer with the true value; it does not provide labels telling the model which source to trust. The model learns the source–time relationship through this answer task and uses it to select the relevant contribution. In the ablation experiments, removing source or temporal information reduced selection to chance. The full representation learned the task, and probes showed attention selecting the temporally reliable source. The numerical answer and its source selection could therefore be evaluated together.

The subsequent numerical work demonstrates the other directions: inferring source from time and content, and time from source and content. It also demonstrates using new observations to improve an answer after the underlying relationship changes. This matters because an evolving pattern commons requires both learning established relationships and responding when new evidence changes them.

5.2 Market-path generation

The Market Transformer applies the architecture to real market data. Each instrument supplies a source identity, its observed value supplies content, and its timestamp locates the observation. The model learns from the relationships within and across these records.

To generate a path, the model is asked for a source's value at successive times. The available context can be held fixed while the requested time moves through a missing interval or beyond the observed record. The resulting sequence is a generated market trajectory that can be plotted against the real path.

The model generated market trajectories from real observations by answering successive source-and-time queries. This extends the proof from individual source selections or missing-value answers to generation over a changing real-world record. The learned representation carries enough structure to express a source's behavior through time.

5.3 Policy decisions

The learned market model was also embedded directly in a policy trained with proximal policy optimization (PPO) [8]. The policy reads its internal representations together with the current portfolio and a memory of previous observations and decisions, then learns whether to buy, sell or hold, and how much to trade, from the consequences of those actions.

The policy performed well across varied market conditions, learning different behavior for pursuing gains and limiting losses. This demonstrates practical use of the learned representations in portfolio decisions, alongside the Market Transformer's demonstration of generation.

These results establish complementary capabilities of the implemented architecture: learning relationships, generating real paths, and using the learned representation to act. The technical foundation of the broader proposal is therefore a functioning numerical system.

6 Next steps

6.1 Training a text model

The immediate next step is to train a small text version of the architecture for Iris on attributed, time-stamped contributions. This requires a text pipeline that retains source identity and time, training infrastructure, and an output mapping for text. The numerical implementation supplies the source–time–content mechanism to be adapted and evaluated in that setting.

Rhizome supplies a persistent, attributed record in which people and agents can connect contributions, revisit earlier reasoning, and continue inquiry. Iris is intended to learn and generate from the relationships within that evolving record, with Ŧrust supplying attention over source, time, and content. An early hypothesis, later evidence, and a widely circulated summary can then be represented as related contributions with distinct authors and temporal positions.

The intended capability is to learn how contributions develop through a pattern commons: who introduces an idea, who tests or corrects it, who connects it to another field, and who helps others understand it. Conflicting accounts remain associated with their sources and times, allowing the model to represent disagreement and its development.

6.2 Evaluating source and temporal learning

Initial evaluation should compare the full text model with content-only models and versions with source or time removed. Controlled tasks can vary a source's reliability across subjects and over time, providing known outcomes against which to evaluate the answer and the contributions used to produce it. Later observations can test whether the model revises its reliance on a source when new evidence changes the relevant relationship.

A further task is to test whether earlier contributions anticipate patterns that subsequently appear across other contributors' work, and whether that relationship recurs in held-out periods. This makes being ahead of the curve a concrete temporal learning problem. Predicting the spread of a belief and establishing its correctness remain distinct evaluation tasks: later uptake measures the former, while subsequent evidence and the chosen learning objective determine the latter.

These evaluations connect the demonstrated numerical operations to learning from the development of collective belief and understanding. They also provide a basis for assessing the separate roles of introducing, testing, correcting, and communicating an idea, rather than assigning all credit to its most recent or most widely repeated expression.

6.3 Extending provenance to generative media

The longer-term media extension carries source and time from training material into the representations used during generation. The proposed provenance record follows contributions through that computation, including intermediate states that combine information from several sources. Extending the readout in Section 4 therefore requires preserving and tracing those relationships across the operations that produce the final work.

Different contributions may concern subject, technique, and composition in an image, or phrasing, rhythm, and production in music. Temporal information distinguishes an early introduction from a later adaptation or repetition. Human direction also contributes: selecting references, arranging their relationship, rejecting outputs, and revising a work changes what is generated. Recording these interventions alongside the model's source-and-time readout would support analysis of the director's contribution.

An illustrative case developed in *Ŧrust Across Dimensions* [9] combines pop-art banana imagery associated with Andy Warhol with the caption "This is not art," echoing René Magritte's "This is not a pipe." The proposed analysis would distinguish those visual and conceptual contributions within the generation process. Another artist's use of a banana would not, by itself, establish that artist's involvement in the output. Attribution concerns the source-aware computation and the history it uses.

7 Conclusion

Ŧrust extends transformer attention by representing source and time alongside content. Its numerical implementations demonstrate that these relationships support inference, generation from real market observations, and practical portfolio decisions. Source-indexed attention weights and grouped value vectors make the allocation and combination of information inspectable within an attention operation.

The next development step is a text model trained and evaluated on attributed contributions through time. This extends a functioning numerical foundation toward Iris: a model that can learn from the development of a pattern commons while retaining the sources and temporal relationships that make those contributions intelligible. The same architecture provides the basis for investigating provenance and credit within generative media.

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, et al. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017. https://arxiv.org/abs/1706.03762

[2] S. Amir, B. C. Wallace, H. Lyu, P. Carvalho, and M. J. Silva. Modelling context with user embeddings for sarcasm detection in social media. *Proceedings of the 20th SIGNLL conference on computational natural language learning*, pp. 167–177, 2016. https://aclanthology.org/K16-1017/

[3] J. Li, M. Galley, C. Brockett, G. P. Spithourakis, J. Gao, and B. Dolan. A persona-based neural conversation model. 2016. https://arxiv.org/abs/1603.06155

[4] S. M. Kazemi, R. Goel, S. Eghbali, J. Ramanan, J. Sahota, S. Thakur, et al. Time2Vec: Learning a vector representation of time. 2019. https://arxiv.org/abs/1907.05321

[5] G. D. Rosin and K. Radinsky. Temporal attention for language models. *Findings of the association for computational linguistics: NAACL 2022*, pp. 1498–1508, 2022. https://aclanthology.org/2022.findings-naacl.112/

[6] J. Ash. Source embeddings. 2026. https://www.speakerjohnash.com/literature/source-embeddings.pdf

[7] J. Ash. Timestamp embeddings. 2026. https://www.speakerjohnash.com/literature/timestamp-embeddings.pdf

[8] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. 2017. https://arxiv.org/abs/1707.06347

[9] J. Ash. Ŧrust across dimensions: Provenance as attention over source, time, and content. 2026. https://www.speakerjohnash.com/literature/trust-across-dimensions-provenance-as-attention-over-source-time-and-content.html