Source Embeddings

John Ash

Independent researcher

First published 28 June 2026; revised 12 November 2026

Abstract

A source embedding gives a contributor a learned representation that is shared across their contributions. Its basic structure is the same as a word embedding: a matrix indexed by discrete identities and optimized through a downstream learning objective. The difference is what the index denotes. Content representations describe what was contributed; source representations identify who or what contributed it. This paper defines that construction, relates it to prior work on user and speaker embeddings, and specifies its implementation in Ŧrust. The implemented module combines an embedding lookup with a shared learned projection. Source, time, content, and an explicit source–time interaction then enter ordinary transformer attention. Numerical source-selection tasks, learned market representations, and their use in a trading policy demonstrate the construction in working systems. These mechanisms provide a basis for learning contextual reliability and recording source-indexed attention, with extensions to language, provenance, and contributor credit.

1 Introduction

Two people can make the same claim while bringing different histories of evidence, experience, and reliability. Two instruments can report the same measurement while having different calibration histories. A model that receives only the content sees the same input in both cases. A model that also receives source identity can learn when those histories matter.

A source embedding supplies that identity as a learned vector. Every contribution from the same registered source addresses the same row of an embedding matrix. The row is updated as the model learns, connecting observations that might otherwise be treated as unrelated examples. A source can be a person, publication, instrument, asset, or another model. The definition depends on the entity that the application needs to distinguish.

This construction extends a familiar mechanism. A word embedding assigns a learned vector to a vocabulary item; a source embedding assigns one to a contributor. Both turn discrete identities into representations that participate in the model's existing computation. Source identity therefore becomes available wherever the model compares, combines, or retrieves information.

The contribution of this paper is to make that symmetry explicit and to describe how it is used in the current Ŧrust systems. Source embeddings operate alongside content and absolute-time representations, allowing attention to learn relationships among **what was contributed, who contributed it, and when**. The wider architecture is developed in the accompanying Ŧrust white paper [1].

2 Related work

Learned word representations established the basic pattern: a discrete vocabulary index selects a parameter vector, and a learning objective shapes the relationships among those vectors. Neural language modeling [2], distributed word representations [3], and GloVe [4] provide different training procedures for this general class of object.

The corresponding construction for people is also established. Amir et al. learn user representations to supply context for sarcasm detection [5]. Li et al. incorporate speaker embeddings into a dialogue model to represent persistent characteristics of a speaker [6]. Yang and Eisenstein use representations derived from a social

network to condition attention over sentiment classifiers [7]. These studies differ in their training signals and use of social information, but each makes an author's identity available as a vector alongside linguistic content.

Ŧrust uses this established representation within transformer attention [8]. Its source vectors are learned jointly with the numerical model, combined explicitly with time, and used in tasks where the relevant contribution depends on the source and temporal context. The embedding lookup is conventional. Its role in a joint account of source, time, and content is the focus here.

3 Definition and learning

3.1 A persistent index and a learned representation

Let $\mathcal{S} = \{1, \dots, N\}$ be a registry of sources and let d be the embedding width. The parameter matrix $E \in \mathbb{R}^{N \times d}$ contains one row per source. Looking up source s returns that row, written as a column vector $e_s \in \mathbb{R}^d$.

The current `SourceEmbedding` module applies a shared learned affine projection after the lookup:

$$e_s = E[s], \qquad z_s = We_s + b. \tag{1}$$

The projection matrix W and bias b are shared across all sources. The module is implemented as `nn.Embedding` followed by `nn.Linear`. Its output z_s is the source representation supplied to the rest of the model.

The persistent object is the source's registry entry. Its parameters can change during training, just as a word's embedding changes while its vocabulary index remains the same. At a fixed model state, all observations assigned to the same source receive the same source representation. Their content and timestamps supply the differences among individual observations.

For a word embedding, the lookup index denotes a token. For a source embedding, it denotes a contributor. Both can use the same width and initialization procedure, but those choices are implementation parameters rather than requirements of the definition. The numerical Ŧrust implementation uses compatible widths so that source and time can interact directly.

3.2 Learning from contributions

Suppose the model is trained with a loss $\mathcal{L}$ computed from its output. Backpropagation updates the source row through the computation that used it. If several observations in a training example share source s, their gradients accumulate at that row:

$$\frac{\partial \mathcal{L}}{\partial e_s} = \sum_{i : s_i = s} W^\top \frac{\partial \mathcal{L}}{\partial z_{s_i}}. \tag{2}$$

The shared projection is trained at the same time. The learning signal therefore determines which distinctions among contributors become useful. A dialogue objective can emphasize speaking style. A prediction objective can emphasize how a source's observations help forecast an outcome. A task involving changing reliability can make source identity useful in different ways at different times.

The source vector is consequently a representation of identity under a learned task, rather than a manually assigned reputation score. It can support many contextual relationships. The same source may be relevant to one question and unhelpful for another because the surrounding content, time, and query also enter the computation.

3.3 Geometry of the source population

Once trained, the source representations form a geometry that can be inspected through similarity, nearest neighbors, or clustering. In the current implementation, the operational geometry is the set of projected vectors z_s, because those are the vectors attention receives. The raw rows e_s remain useful for parameter inspection, but their pairwise distances need not match distances after the shared projection.

The resulting geometry reflects the task and training data. Sources whose contributions support similar predictions can develop related representations. This is the source-space counterpart of learning related word representations from related linguistic contexts. It is an empirical property to examine in the trained model, rather than a semantic label assigned to an axis in advance.

4 Composition with time and content

An observation has content c_i, timestamp τ_i, and source identity s_i. Let v_i be its content representation and u_i its time representation, projected to the source width. The implemented numerical construction is

$$x_i = [v_i; u_i; z_{s_i}; u_i \odot z_{s_i}]. \tag{3}$$

The brackets indicate concatenation, and $\odot$ denotes elementwise multiplication. The first three parts preserve the observation's distinct coordinates. The fourth gives subsequent layers an explicit interaction between source and time. It lets the representation of a source's contribution vary with the moment while retaining the same source identity.

The time vector is produced by a learned absolute-time encoder with trend and periodic branches, described in *Timestamp Embeddings* [9]. In the current Market Transformer and embedded policy, the content part combines learned projections of numerical value and return. These representations then pass through contextual attention and further learned computation.

This is an extension of the information supplied to attention. It uses the ordinary query, key, and value projections of a transformer. Attention can compare observations using their content, their source, their time, and the interactions learned from those inputs. No separate rule is needed to turn a source identifier into a hand-written trust score.

The source–time product is particularly natural when reliability changes periodically. For example,

$$\sin(\omega\tau + \phi_s) = \sin(\omega\tau)\cos(\phi_s) + \cos(\omega\tau)\sin(\phi_s). \tag{4}$$

This identity expresses a relationship involving time and source phase through products of time-dependent and source-dependent terms. The implemented interaction makes such cross-terms available directly. Learned downstream layers determine how to use them for the task.

5 Use in the numerical systems

5.1 Learning which contribution to read

The controlled source-selection task asks a concrete question: can a model learn whose contribution to use when the relevant source changes with time? Several sources offer numerical values at each timestamp. A time-dependent rule determines which source supplies the true value; the other sources supply independent noise from the same marginal distribution. A value's magnitude alone therefore does not identify the expert.

Each example contains source-labelled observations at several timestamps. The model processes their joint representations and uses timestamp queries to produce answers. Training compares those numerical answers with the true values. It does not supply an expert-source label to the loss. Source and time become useful because they help the model choose the contribution that answers the task.

The recorded experiments show that the full representation learns this relationship, while configurations missing source, time, or both remain near chance at identifying the expert's offered value. Those reductions alter the available representation; removing time also replaces timestamp-query attention with pooling. They establish the importance of source and time for this task. The explicit source–time product is part of the successful full construction.

The output and attention are inspected separately. The numerical-output check asks which offered value is closest to the answer. The attention inspection groups the read weights by source and asks which source receives the largest share. This makes it possible to examine the relationship between learning to attend to a contributor and learning to reproduce that contributor's value.

These experiments are part of the design's development record. Their earlier timestamp-normalization variant is distinct from the absolute-time transport now used in the current systems. The source lookup, learned projection, and joint source–time construction carry through to the current implementation.

5.2 Market representations and practical decisions

The Market Transformer applies the construction to real observations from multiple financial instruments and registered predictive sources. Content, timestamp, and source are retained together. The model learns relationships across those observations and generates numerical trajectories in response to source-and-time requests [1].

Inspection of its learned source geometry gives a concrete example of what the representation can capture. In a recorded market checkpoint, projected source vectors grouped related gold and semiconductor instruments, although sector labels were not supplied to the optimization. The analysis compared embedding relationships with realized return relationships. This connects the geometry to the behavior the model actually learned from, rather than assigning human meanings to arbitrary coordinates.

The representation is also used inside the embedded trading policy. The policy retains the Market Transformer's source and time modules and reads the resulting latent field alongside portfolio and recurrent state. The learned representation thereby participates in practical buy, sell, hold, and sizing decisions. End-to-end training can adapt those modules through the policy's objective; serving uses the saved learned parameters.

The current loading path preserves the source registry associated with the checkpoint, including extensions to the registered population. The mathematical operation remains the lookup and shared projection in Equation 1. Its MPS training path can implement the lookup with a one-hot matrix product to obtain the same representation and parameter gradients.

6 Source-indexed attention and provenance

Source identity gives an attention read an explicit contributor index. For one attention head, let q be a query, k_i a key, and w_i the value vector at eligible context position i. Standard attention computes

$$\alpha_i(q) = \text{softmax}_i\left(\frac{q^\mathsf{T} k_i}{\sqrt{d_k}}\right), \qquad r(q) = \sum_i \alpha_i(q) w_i, \tag{5}$$

where d_k is the key width. Because the context positions retain source identifiers, the same operation can be grouped by source:

$$A_s(q) = \sum_{i:s_i=s} \alpha_i(q), \qquad R_s(q) = \sum_{i:s_i=s} \alpha_i(q) w_i, \qquad r(q) = \sum_s R_s(q). \tag{6}$$

These quantities answer two related questions. A_s records how much attention the read allocates to source-labelled positions. R_s records the vector contribution from those positions to that read. The latter preserves information that a scalar weight alone does not express.

The computation supplies a precise, local receipt. When the attended representations have already combined observations, their earlier relationships also matter to the provenance of the final answer. Following those relationships through the full model is how local receipts can contribute to an account of generation. Merely adding an embedding table does not by itself perform that tracing.

Within this construction, contextual trust is expressed through what the model learns to use for a particular request. Its allocation can change with the topic, moment, and available evidence. Source identity makes that allocation addressable and inspectable; the task and its outcomes teach the model which relationships help.

7 Next steps

The next development step is to train a text model that retains source and time alongside linguistic content. A passage would carry its author's registered identity and its time, while tokens would still retain the positional information needed for language. The numerical architecture supplies the implemented foundation for this extension.

The objective is to learn from the development of contributions over time: who introduced an idea, who tested it, who improved it, and whose claims proved useful in later contexts. Such training requires examples that connect contributions with subsequent evidence and outcomes. It also requires preserving source relationships through contextual processing so that the model's use of those contributions can be examined.

The same foundation is relevant to creative credit. A source-indexed account of generation could distinguish originating, refining, combining, and directing contributions. Developing that account in language and media means carrying the relationships through the generative process and evaluating the resulting attribution, building on the local attention decomposition already available numerically.

A further design property is uniform representational capacity: every registered contributor receives the same number of source-specific parameters. This provides a concrete primitive for representative aggregation. Sampling frequency, training objectives, and the evidence associated with each contributor still determine how those parameters are learned and used. Equal row size and learned influence are distinct choices that can be addressed explicitly.

8 Conclusion

Source embeddings apply a standard learning mechanism to a consequential index: the contributor. A stable registry entry connects a source's observations, while a learned vector and shared projection make that identity available to the model's existing operations. Combined with time and content, the representation supports relationships that anonymous observations cannot express directly.

Ŧrust uses this construction in numerical source selection, real-market representations, and an embedded decision policy. It also makes individual attention reads decomposable by source. These implemented mechanisms establish a practical foundation for extending source-aware learning to language and for making the use of contributors' work inspectable. Word embeddings represent what is said; source embeddings preserve who contributed it. Both belong in a model intended to learn from an evolving body of collective work.

References

[1] J. Ash. Ŧrust: Learning where patterns come from and how they change through time. 2026. https://www.speakerjohnash.com/literature/trust-white-paper.pdf

[2] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. A neural probabilistic language model. *Journal of Machine Learning Research*, 3, pp. 1137–1155, 2003. https://jmlr.org/papers/v3/bengio03a.html

[3] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*, 26, 2013. https://arxiv.org/abs/1310.4546

[4] J. Pennington, R. Socher, and C. D. Manning. GloVe: Global vectors for word representation. *Proceedings of EMNLP*, pp. 1532–1543, 2014. https://aclanthology.org/D14-1162/

[5] 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/

[6] 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

[7] Y. Yang and J. Eisenstein. Overcoming language variation in sentiment analysis with social attention. *Transactions of the Association for Computational Linguistics*, 5, pp. 295–307, 2017. https://aclanthology.org/Q17-1021/

[8] 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

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