Event2Vec: A Geometric Approach to Learning Composable Representations of Event Sequences

Antonin Sulc
ASULC@LBL.GOV
Lawrence Berkeley National Laboratory
Berkeley, CA, U.S.A.

Abstract

The study of neural representations, both in biological and artificial systems, is increasingly revealing the importance of geometric and topological structures. Inspired by this, we introduce Event2Vec, a novel framework for learning representations of discrete event sequences. Our model leverages a simple, additive recurrent structure to learn composable, interpretable embeddings. We provide a theoretical analysis demonstrating that, under specific training objectives, our model's learned representations in a Euclidean space converge to an ideal additive structure. This ensures that the representation of a sequence is the vector sum of its constituent events, a property we term the linear additive hypothesis. To address the limitations of Euclidean geometry for hierarchical data, we also introduce a variant of our model in hyperbolic space, which is naturally suited to embedding tree-like structures with low distortion. We present experiments to validate our hypothesis and demonstrate the benefits of each geometry, highlighting the improved performance of the hyperbolic model on hierarchical event sequences.

1. Introduction

The convergence of neuroscience and machine learning has highlighted the critical role of geometry in shaping neural representations Bronstein et al. (2017). In the brain, neural activity often unfolds on low-dimensional manifolds, reflecting the underlying structure of tasks and environments Kriegeskorte and Kievit (2013). Similarly, in artificial intelligence, principles like equivariance and compositionality are key to developing generalizable and interpretable models Cohen and Welling (2016). This paper contributes to this effort by investigating how a simple geometric prior can lead to highly structured and interpretable representations of sequential data.

In this paper, we address the challenge of learning representations for sequences of discrete events by introducing Event2Vec, a model designed to learn geometrically structured and sequentially compositional representations. Our central hypothesis, which we term the *linear additive hypothesis* Mikolov et al. (2013a), is that the representation of an entire event history can be modeled as the vector sum of the embeddings of its constituent events. While based on addition, this structure enables a rich vector arithmetic, allowing for both the composition (via addition) and decomposition (via subtraction) of event trajectories. This high degree of interpretability allows for reasoning about entire trajectories; for instance, the displacement vector between first job and promotion (a subtraction) can represent the abstract concept of 'career progression'. Such a structure allows us to analyze a complex event trajectory by deconstructing it into its ordered, constituent steps, providing a clear path towards geometric mechanistic interpretability for sequential data.

We present two variants of our model: one operating in a standard Euclidean space and another in a hyperbolic space. Hyperbolic geometry is particularly well-suited for embedding hierarchical or tree-like structures with low distortion Nickel and Kiela (2017).

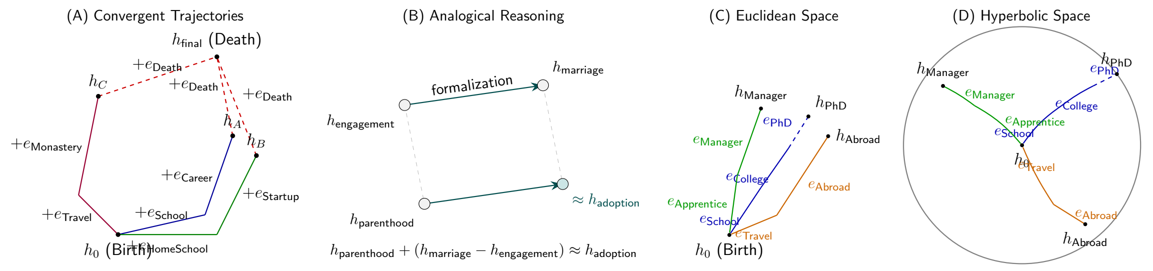

Figure 1: **Geometric Properties of Event2Vec Embeddings.** The model learns to represent event sequences as trajectories in a vector space. **(A)** Probable event sequences form trajectories where consecutive event vectors are directionally aligned. For example, two distinct life paths ('homeschool' → 'career' and 'startup' → 'career') converge towards a similar state ('death') by adding event vectors that follow a logical progression. **(B)** The additive structure enables analogical reasoning through vector arithmetic. The "formalization" vector learned from the ‘engagement‘ to ‘marriage‘ transition can be applied to ‘parenthood‘ to correctly identify the parallel concept of ‘adoption‘. **(C)** Euclidean space provides a simple, flat geometry for these trajectories. **(D)** Hyperbolic space, with its exponential volume, is better suited for capturing the hierarchical branching often found in complex event data, preventing the 'crowding' of distinct paths.

2. Related Work

The concept of vector space embeddings for words, such as Word2Vec Mikolov et al. (2013b) and GloVe Pennington et al. (2014), demonstrated the power of capturing semantic relationships through vector arithmetic. These models are trained on local co-occurrence statistics within a fixed context window. However, their reliance on local co-occurrence statistics within a fixed context window makes them unsuitable for modeling the long-range, ordered dependencies found in event sequences. In contrast, our model's recurrent architecture and novel reconstruction loss are designed to explicitly capture this directed, temporal structure, enforcing sequential compositionality over the entire event history.

Our work is related to methods for learning node representations in graphs, such as DeepWalk Perozzi et al. (2014) and Node2Vec Grover and Leskovec (2016). These methods generate sequences of nodes through random walks on the graph and then use them to learn embeddings. However, these approaches treat the generated walks as unordered contexts. They are designed to capture neighborhood structure rather than the explicit, directed, and temporal order of events in a sequence, which is the primary focus of our model.

Recurrent Neural Networks (RNNs) and their variants, like Long Short-Term Memory (LSTM) Hochreiter and Schmidhuber (1997) and Gated Recurrent Units (GRUs) Cho et al. (2014), are the standard for modeling sequential data. While powerful, their complex, non-linear dynamics, involving gating mechanisms and matrix multiplications, can lead to representations that are difficult to interpret. Other powerful sequence models like Neural Temporal Point Processes Shchur et al. (2021); Mei and Eisner (2017); Du et al. (2016) also learn embeddings from event histories, their complex, non-linear representations are optimized for temporal prediction and are not designed to support the kind of simple, interpretable vector arithmetic that our model explicitly enforces.

The idea of using geometric priors in deep learning termed Geometric Deep Learning Bronstein et al. (2017), has gained traction, including work on graph neural networks, equivariant models, and the use of non-Euclidean geometries. Our work is a specific instance of this paradigm, imposing a strong additive prior on a sequence model. Our hyperbolic prototype is directly inspired by recent advances showing that hyperbolic spaces are particularly effective for embedding hierarchical data. Their exponential volume growth naturally accommodates tree-like structures with low distortion, a property first leveraged for representation learning in Poincaré embeddings Nickel and Kiela (2017) and later extended to neural network models that operate directly within curved geometry Ganea et al. (2018).

Our approach introduces a novel contribution by bridging the gap between purely sequential models and context-based embedding methods. While RNNs capture order and methods like Word2Vec capture context, our model is, to our knowledge, the first to explicitly unify both within a simple, geometrically interpretable framework. The core novelty lies in our training objective, which learns representations that are not just contextually similar but are also compositionally sequential. Specifically, our loss function encourages the model to place likely subsequent events linearly along a trajectory in the embedding space, while positioning semantically unrelated or unlikely events in orthogonal directions. This creates a much richer geometric structure where the vector displacement between events is directly tied to their sequential probability, a property not explicitly enforced by prior methods.

3. The Event2Vec Model

We model a sequence of events $S = (s_1, s_2, ..., s_T)$, where each event s_t is drawn from a vocabulary of event types. Our goal is to learn an embedding vector $e_i \in \mathbb{R}^d$ for each event type i, and a hidden state vector $h_t \in \mathbb{R}^d$ that represents the history of events up to time t.

3.1. Model Architecture

The core of the Event2Vec model is its additive state update mechanism, where the event embedding e_{s_t} encodes a transition that is applied to the previous state h_{t-1}. We propose two geometric variants: a Euclidean model using standard vector addition, and a Hyperbolic model using Möbius addition, which is better suited for hierarchical data.

Euclidean Model In the Euclidean variant, the update rule is a pure vector addition. However, simple additivity can be numerically unstable, as the magnitude of the hidden state vector can grow without bound over long sequences. To ensure stable training, we employ a regularization technique. After the additive update, we clip the L2 norm of the resulting hidden state vector to a maximum value. This preserves the direction of the vector, and thus the additive semantics, while preventing its magnitude from causing training instabilities.

$$h_t = h_{t-1} + e_{s_t} \quad \text{(with norm clipping)} \tag{1}$$

The initial hidden state h_0 is a zero vector. The model predicts the next event s_{t+1} via a linear decoder followed by a softmax function:

$$P(s_{t+1}|h_t) = \text{softmax}(W_{dec}h_t + b_{dec}) \tag{2}$$

Hyperbolic Model For the hyperbolic variant, we operate in the Poincaré ball model from Ganea et al. (2018), a conformal model of hyperbolic space that represents points within a unit ball. The additive update is replaced by Möbius addition $\oplus_c$, the natural generalization of addition to hyperbolic space with curvature $c < 0$. It is defined as:

$$x \oplus_c y = \frac{(1 + 2c\langle x, y\rangle + c\|y\|^2)x + (1 - c\|x\|^2)y}{1 + 2c\langle x, y\rangle + c^2\|x\|^2\|y\|^2} \tag{3}$$

This operation ensures that the addition of two vectors within the Poincaré ball results in another vector that also lies within the ball. The update rule is:

$$h_t = h_{t-1} \oplus_c e_{s_t} \tag{4}$$

To make a prediction, we must interface the hyperbolic representation with a standard Euclidean classifier. We do this by mapping the hyperbolic state h_t to the tangent space at the origin (a Euclidean space) via the logarithmic map, $\log_{0,c}(\cdot)$, before applying the linear transformation. This is a standard and principled approach for decoding from hyperbolic representations.

3.2. Training Objective

The model is trained to minimize a composite loss function $\mathcal{L}_{total}$, which consists of three components designed to enforce our desired geometric properties:

1. **Prediction Loss ($\mathcal{L}_{pred}$):** A standard cross-entropy loss for predicting the next event in the sequence. This ensures the model learns to capture the sequential dependencies in the data.

$$\mathcal{L}_{pred} = -\sum_{t=1}^{T-1} \log P(s_{t+1}|h_t) \tag{5}$$

2. **Reconstruction Loss ($\mathcal{L}_{recon}$):** This loss is central to our linear additive hypothesis. It ensures that the event embedding can be "subtracted" from the hidden state to perfectly recover the previous state. This explicitly enforces the algebraic group structure of addition and its inverse.

$$\mathcal{L}_{recon} = \sum_{t=1}^{T} \|(h_t - e_{s_t}) - h_{t-1}\|_2^2 \tag{6}$$

   For the hyperbolic case, this becomes $\mathcal{L}_{recon} = \sum_t d_c(h_t \oplus_c (-e_{s_t}), h_{t-1})^2$, where d_c is the Poincaré distance.

3. **Consistency Loss ($\mathcal{L}_{consist}$):** To ensure robustness and a smooth embedding space, we use dropout as a stochastic perturbation Gao et al. (2021). We run the same input through the model twice and penalize differences in the resulting hidden states.

This encourages the model to be insensitive to small perturbations, leading to a more stable and generalizable representation space.

$$\mathcal{L}_{consist} = \sum_{t=1}^{T} \|h_t^{(1)} - h_t^{(2)}\|_2^2 \tag{7}$$

where $h_t^{(1)}$ and $h_t^{(2)}$ are the hidden states from two separate forward passes with the same input but different dropout masks.

The total loss is a weighted sum of these components:

$$\mathcal{L}_{total} = \mathcal{L}_{pred} + \lambda_{recon}\mathcal{L}_{recon} + \lambda_{consist}\mathcal{L}_{consist} \tag{8}$$

4. Theoretical Justification for Additivity

Here, we justify that for the Euclidean Event2Vec model, minimizing our composite loss function forces the recurrent update to converge to an ideal additive form. This provides a theoretical foundation for our claim that the model learns composable representations.

Theorem 1 (Justification for Ideal Additivity) *Let the hidden state update be defined as $h_t = f(h_{t-1}, e_{s_t})$, where f is a function parameterized by the neural network. Minimizing the reconstruction loss $\mathcal{L}_{recon} = \sum_t \|f(h_t, -e_{s_t}) - h_{t-1}\|_2^2$ with respect to the parameters of f drives f to approximate the linear additive function $h_t = h_{t-1} + e_{s_t}$.*

Proof The reconstruction loss, $\mathcal{L}_{recon}$ reaches its minimum of zero only when its argument is zero for all steps. This imposes the following constraint on the learned update function f:

$$f(f(h_{t-1}, e_{s_t}), -e_{s_t}) = h_{t-1}$$

This equation dictates that the update operation must be **perfectly reversible**, the effect of applying an event embedding (e_{s_t}) is undone by applying its additive inverse ($-e_{s_t}$).

The simplest function satisfying this property in a vector space is **linear addition**, $f(h, e) = h + e$, which an optimizer will converge to. ■

It is important to note that while the reconstruction loss enforces this reversible structure, it does not by itself guarantee a meaningful solution. For instance, a trivial solution where all event embeddings are zero ($e_t = \mathbf{0}$) would perfectly satisfy the reversibility constraint. This outcome is prevented by the prediction loss, which compels the model to learn informative, non-zero embeddings to make accurate forecasts about the sequence. While the reconstruction loss enforces the algebraic structure of additivity, the prediction loss imposes the semantic grounding for the embeddings. The two work in tandem.

Theorem 2 (Semantic Grounding via Prediction Loss) *The prediction loss $\mathcal{L}_{pred}$ encourages the inner product of a hidden state h_t and a subsequent event embedding $e_{s_{t+1}}$ to be proportional to their pointwise mutual information (PMI).*

Proof [Sketch] The $\mathcal{L}_{pred}$ provides the semantic grounding for the embeddings by linking them to the statistical properties of the event sequences. The objective is analogous to the Skip-Gram with Negative-Sampling (SGNS) framework Levy and Goldberg (2014), which has been shown to implicitly factorize a word-context matrix where each cell is the shifted Pointwise Mutual Information (PMI). The optimal solution for the dot product of a word vector and its context vector under the SGNS objective is $\boldsymbol{w}^T\boldsymbol{c} = \text{PMI}(w, c) - \log k$.

By analogy, in our model, the next event s_{t+1} acts as the "word" and the history vector h_t acts as the "context." Therefore, at the optimum, the model learns embeddings such that $h_t^T e_{s_{t+1}} \approx \text{PMI}(s_{t+1}, h_t) - \log k$. A crucial feature of our model is the linear additive hypothesis, where the history vector is the sum of its constituent event embeddings $h_t = \sum_{i=1}^t e_{s_i}$ (additive hypothesis). Substituting this into the relationship yields:

$$h_t^T e_{s_{t+1}} = \underbrace{\left(\sum_{i=1}^t e_{s_i}\right)^T e_{s_{t+1}}}_{\text{Additive hypothesis}} = \underbrace{\sum_{i=1}^t e_{s_i}^T e_{s_{t+1}}}_{\text{Distributive property}} \approx \underbrace{\text{PMI}(s_{t+1}, \{s_1, \dots, s_t\})}_{\text{by analogy to the SGNS objective}}$$

This result demonstrates that the prediction loss $\mathcal{L}_{pred}$ forces the embeddings to be geometrically arranged such that events statistically likely to follow a sequence are directionally aligned with that sequence's vector. This grounds the learned geometry in the statistical reality of the data. ■

The prediction and reconstruction losses work in synergy to create a space that is both semantically meaningful and structurally coherent. The prediction loss first provides the semantic grounding, arranging the space by ensuring that embeddings for statistically likely event sequences (e.g., 'birth' then 'infancy') are geometrically close. However, this proximity alone does not enforce a specific algebraic rule. The reconstruction loss then acts as a structural constraint, taking this semantically organized space and "cementing" it with the rules of vector arithmetic. By forcing the event update to be perfectly reversible, $\mathcal{L}_{recon}$ ensures that the representation of a sequence is the precise vector sum of its parts (e.g., $h(\text{'birth'}, \text{'infancy'} = e_{birth} + e_{infancy})$, making the learned space truly and interpretably compositional.

Theorem 3 (Hyperbolic Equivalence) *In the Poincaré ball, minimizing the hyperbolic reconstruction loss* $\mathcal{L}_{recon} = \sum_t d_c(h_t \oplus_c (-e_{s_t}), h_{t-1})^2$ *forces the update rule to be Möbius addition.*

Proof The proof follows the same logic as the Euclidean case, but with hyperbolic operations. The distance $d_c(x, y)$ is minimized when $x = y$. Thus, the loss is minimized when $h_t \oplus_c (-e_{s_t}) = h_{t-1}$. In hyperbolic geometry, the inverse of Möbius addition with a vector e is Möbius addition with its negative, $-e$. Let the update rule be $h_t = f_c(h_{t-1}, e_{s_t})$. The loss minimum requires $f_c(f_c(h_{t-1}, e_{s_t}), -e_{s_t}) = h_{t-1}$. The function that satisfies this property in the Poincaré ball is Möbius addition, $f_c(h, e) = h \oplus_c e$. Therefore, minimizing the hyperbolic reconstruction loss forces the model to learn the correct additive compositional rule for that geometry. ■

5. Experiments

To validate our model and its theoretical underpinnings, we propose the following experiments designed to rigorously test our claims and demonstrate the utility of our approach. In all experiments, the loss weights λ were set to 1.

5.1. Life Path Example

To demonstrate the practical utility and interpretability of Event2Vec, we apply it to a synthetic dataset modeling a human life path. This dataset consists of sequences of significant life events, from birth and education to career milestones and retirement. The goal of this experiment is to qualitatively assess whether our model can learn a geometrically coherent representation of a chronological trajectory, a task that requires capturing not just the semantic meaning of events but also their inherent temporal order. Appendix explains the entire scenario.

ID	**Analogy Query** ($A - B + C$)						**Result** (D)	Cosine sim.
1A	‘elementary_school‘	-	‘birth‘	+	‘first_job‘	=	‘late_childhood‘	0.4504
1B	‘death‘	-	‘retirement‘	+	‘graduation‘	=	‘internship‘	0.4090
2A	‘promotion‘	-	‘career_start‘	+	‘first_job‘	=	‘military_service‘	0.5048
2B	‘business_success‘	-	’entrepreneurship’	+	’investment’	=	‘inheritance‘	0.5976
3A	‘marriage‘	-	’engagement‘	+	‘parenthood‘	=	‘adoption‘	0.4653
3B	‘divorce‘	-	‘relationship_challenge‘	+	‘financial_hardship‘	=	‘major_purchase‘	0.3869

Table 1: Analogical reasoning examples demonstrating that vector arithmetic ($A - B + C$) on Event2Vec embeddings captures coherent semantic relationships between life events.

Analogical Reasoning Task: The examples in Table 1 show the model’s ability to learn abstract relationships through analogical reasoning. For example, the query in row **1B** is interpreted as a ’life phase transition,’ mapping the career-to-retirement shift onto the school-to-work transition to yield ‘internship‘ as the model’s result. Other tests reveal the model’s grasp of thematic clustering (row **2B**), where it groups ‘business_success‘ and ‘inheritance‘ as major financial events, and even causal reasoning (row **3B**), where it identifies ‘major_purchase‘ as a common cause of financial hardship.

Geometric Analysis of Embedding This theoretical structure is borne out in our experiments. The model’s learned geometry, shown in Figure 2, reveals distinct life trajectories. A primary path flows from an initialization phase of education (‘birth‘, ‘study_abroad‘) into a dense central hub representing a conventional career and family life (‘marriage‘, ‘leadership_role‘), with logical offshoots for events like ‘divorce‘ and ‘grandparenthood‘. The visualization also captures alternative paths, such as a specialized academic sequence and individualistic pursuits (‘travel‘, ‘volunteer_work‘). Critically, the event ‘death‘ is not a single endpoint but is located centrally, reflecting its status as a shared outcome. However, this static visualization does not convey the full power of Event2Vec; its compositional additivity, which enables arithmetic reasoning on trajectories, is more robustly validated by the quantitative results of the Brown Corpus experiment in the following section.

Comparison of Euclidean vs. Hyperbolic Space: We trained a hyperbolic variant of Event2Vec to better model the inherently hierarchical nature of life paths, leveraging

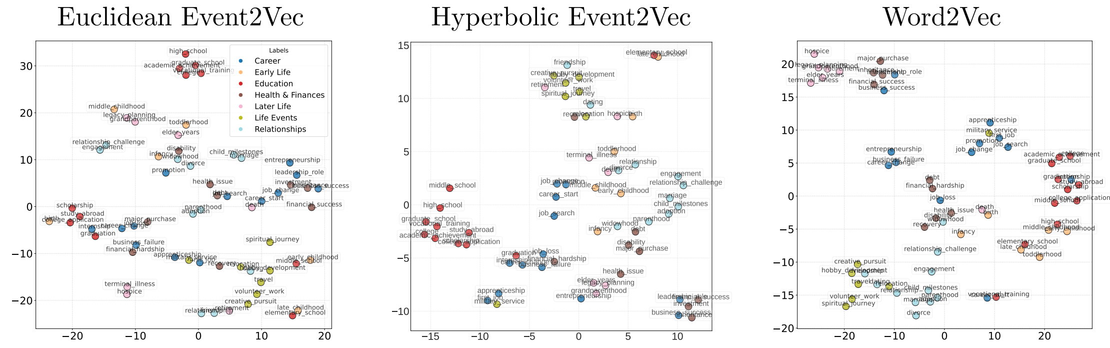

Figure 2: A t-SNE comparison of life path embeddings. **(Left)** The Euclidean Event2Vec model learns a clear chronological trajectory. **(Center)** The Hyperbolic Event2Vec model captures a more powerful hierarchical structure, with life paths branching radially from the ‘birth‘ event. **(Right)** The Word2Vec baseline learns thematic clusters, grouping events by semantic context (e.g., ‘birth‘ and ‘death‘) while failing to capture sequential or hierarchical relationships.

a geometry designed for tree-like data. The resulting embedding space, visualized using Poincaré distances, suggests a branching structure where life events appear to separate into domains like Education, Relationships, and Career. We hypothesize that this structure may be more effective at representing the divergent nature of life choices than a standard Euclidean trajectory, while still preserving the temporal structure.

5.2. Brown Corpus

To validate our model’s ability to capture linguistic structure, we conducted an unsupervised experiment on the Brown Corpus Bird et al. (2009). We trained Event2Vec and a Word2Vec baseline on raw sentences without any grammatical labels. Subsequently, we evaluated the models by composing vector representations for specific Part-of-Speech (POS) tag sequences (e.g., ‘AT-JJ-NN‘ for article-adjective-noun) by summing the learned embeddings of corresponding words. This directly tests if the model’s additive property learns meaningful syntactic composition.

The results, visualized in Figure 3, show that Event2Vec successfully organizes these composite vectors into distinct clusters corresponding to their grammatical structure. Notably, it groups similar patterns like ‘AT-JJ-NN‘ and ‘IN-AT-NN‘, a separation that is far less distinct in the Word2Vec baseline. This visual finding is quantitatively supported by a silhouette score of 0.0564, more than double the 0.0215 achieved by Word2Vec. This demonstrates that Event2Vec can discover and represent grammatical regularities from raw text without any explicit supervision.

6. Limitations

The model’s deliberate simplicity and reliance on a additive structure introduce two key limitations. First, the pure additive mechanism is numerically unstable on long sequences, as the hidden state vector can grow without bound (see Figure 12 in Appendix). This necessitates regularization techniques that, in turn, compromise the model’s perfect additivity.

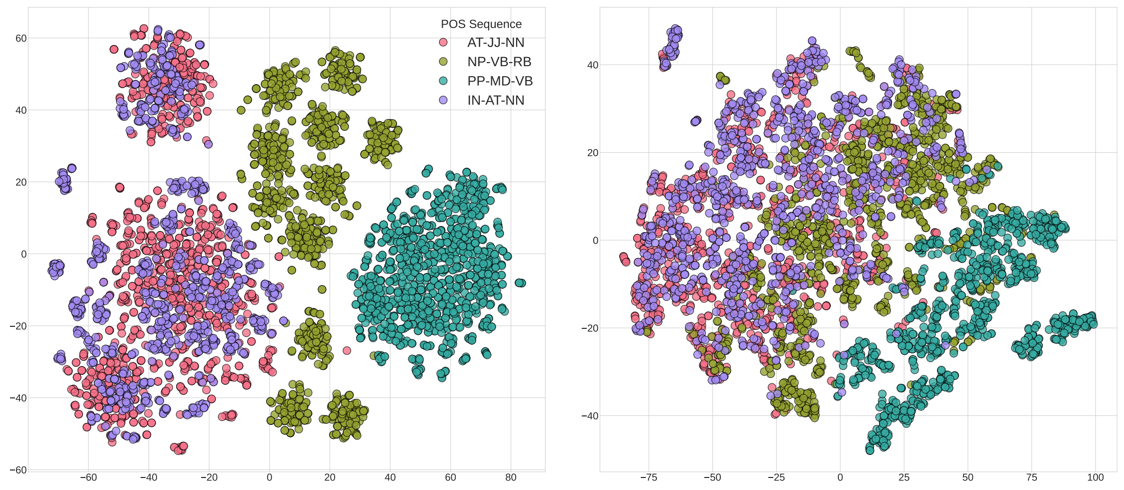

Figure 3: t-SNE visualization of embedded POS tag sequences from the Brown Corpus. Event2Vec (left) produces well-defined clusters corresponding to grammatical structures, correctly grouping similar sequences like ‘AT-JJ-NN‘ and ‘IN-AT-NN‘. The Word2Vec baseline (right) shows significantly weaker separation and overlapping clusters.

Second, the additive prior is a conceptual constraint, as it cannot capture complex, non-linear interactions between events (e.g., the effect of a ’market_crash’ on an ’investment’). Our approach sacrifices the ability to model these complex dynamics in favor of a more interpretable geometric framework.

7. Conclusion

We introduced Event2Vec, a model that learns composable representations of event sequences by enforcing geometric simplicity. Our theoretical analysis provides a justification that our training objective guides the model to learn a Euclidean representation space with an ideal additive structure, a property we term the linear additive hypothesis. This structure is key to the model’s utility, offering a clear path toward mechanistic interpretability by allowing a meaning to be derived from the sum of its parts. A key quantitative result is the model’s outcome on the Brown Corpus, where it discovered and clustered distinct grammatical structures from raw text in a completely unsupervised manner outperforming a Word2Vec baseline. This provides powerful evidence that the learned geometry captures meaningful, compositional semantics.

Recognizing that Euclidean geometry is ill-suited for hierarchical data, we also developed and investigated a hyperbolic variant of our model. The exponential ”room” in hyperbolic geometry is intuitively better suited for embedding tree-like data structures with low distortion, which is a common characteristic of branching paths. Our experiments on synthetic life-path data confirm the utility of this approach, with the hyperbolic variant show a potential to capture hierarchical relationships over Euclidean space.

Finally, we acknowledge the trade-offs inherent in our design. The model’s simplicity, while central to its interpretability, introduces limitations in numerical stability and conceptually restricts it from capturing complex, non-linear event interactions. This work establishes a clear framework for geometrically-grounded, interpretable sequence representation, paving the way for future research into hybrid models that seek to balance this interpretability with the capacity for more complex dynamics.

References

Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.", 2009.

Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Taco S Cohen and Max Welling. Group equivariant convolutional networks. In *International conference on machine learning*, pages 2990–2999. PMLR, 2016.

Nan Du, Hanjun Dai, Rakshit Trivedi, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and Le Song. Recurrent marked temporal point processes: Embedding event history to vector. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1555–1564, 2016.

Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. Hyperbolic neural networks. In *Advances in neural information processing systems 31*, pages 5350–5360, 2018.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*, 2021.

Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, 2016.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Nikolaus Kriegeskorte and Rogier A Kievit. Representational geometry: integrating cognition, computation, and the brain. *Trends in cognitive sciences*, 17(8):401–412, 2013.

Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems 27*, pages 2177–2185, 2014.

Hongyuan Mei and Jason M Eisner. The neural hawkes process: A neurally self-modulating multivariate point process. *Advances in neural information processing systems*, 30, 2017.

Tomas Mikolov, Quoc V Le, and Ilya Sutskever. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*, 2013a.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems 26*, pages 3111–3119, 2013b.

Maximilian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. In *Advances in neural information processing systems 30*, pages 6338–6347, 2017.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710, 2014.

Oleksandr Shchur, Ali Caner Türkmen, Tim Januschowski, and Stephan Günnemann. Neural temporal point processes: A review. *arXiv preprint arXiv:2104.03528*, 2021.

Experiments

Life Scenario

The following figures show states and transitions of the life scenario as described by our model. We begin with Figure 4 which provides an overview of the individual blocks shown in further detail below.

We see the education in Figure 6, which maps out the paths through the educational system. From there, Figure 7 details the professional life. Running parallel to this, Figure 11 illustrates the transition to states of e.g. finding partners and building a family. Personal choices shown in Figure 10 can alter this path, while the critical interplay of wellness and wealth is detailed in Figure 8. The scenario transition concludes in Figure 9, which shows the transition from retirement into the elder years. In each of these detailed figures, the arrows show the chance of moving from one event to the next.

Dataset Generation

The synthetic Life Path dataset was generated using a stochastic process designed to simulate realistic life trajectories. The generation follows a guided random walk on a state transition graph, where nodes represent life events.

The process for generating a single sequence is as follows:

- **Initialization**: Each sequence begins at the predefined initial state, ‘birth‘.
- **Probabilistic Transition**: At each step, the next event is chosen based on the current event. For a given event, there is a predefined dictionary of possible subsequent events, each with an assigned probability weight. The next event is selected by sampling from this distribution.
- **Stochastic Exploration**: To ensure diversity in the generated sequences and model less common life paths, a 10% chance of a random transition is introduced at every step. In this case, instead of following the weighted probabilities, the model selects the next event uniformly at random from the entire vocabulary of possible life events.
- **Termination**: The sequence generation continues until the terminal state, ‘death‘, is reached or a maximum sequence length of 16 events is exceeded. If the maximum length is reached, the ’Death’ event is appended to formally conclude the trajectory.

This procedure was used to generate 10,000 unique life path sequences, which formed the training set for the experiments described in Section 5.1.

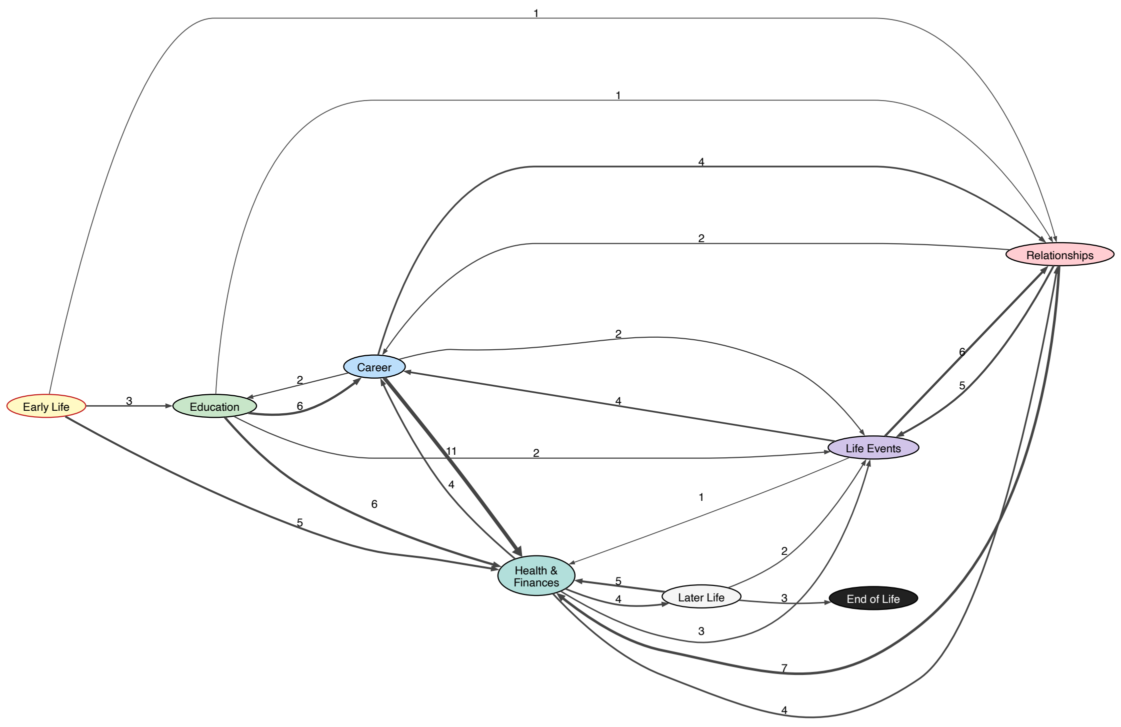

Figure 4: A high-level overview of the life model, where each node represents a major life stage. The directed edges indicate the flow between stages, with the label and thickness of each edge representing the total number of distinct transition paths. This illustrates the central roles of the **Career** and **Health & Finances** stages as highly interconnected hubs in a person's life journey.

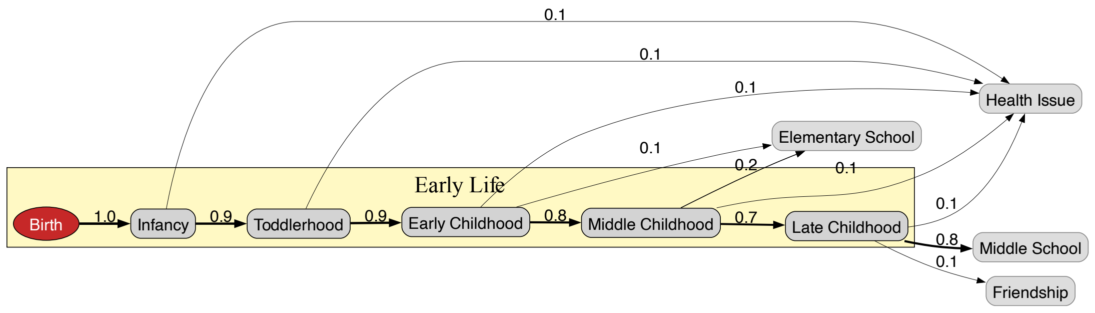

Figure 5: The Early Life stage, showing the foundational years from the ‘birth‘ event through childhood. The diagram shows a strong, linear progression through key developmental milestones, from ‘infancy‘ to ‘Late Childhood‘, with high transition probabilities indicating a canonical path. The stage concludes with transitions to later life phases such as ‘Middle School‘ and the formation of ‘Friendship‘. Notably, the model also captures the persistent, low-probability risk of a ‘Health Issue‘ occurring at any point during this period, demonstrating its ability to model both sequential and parallel events.

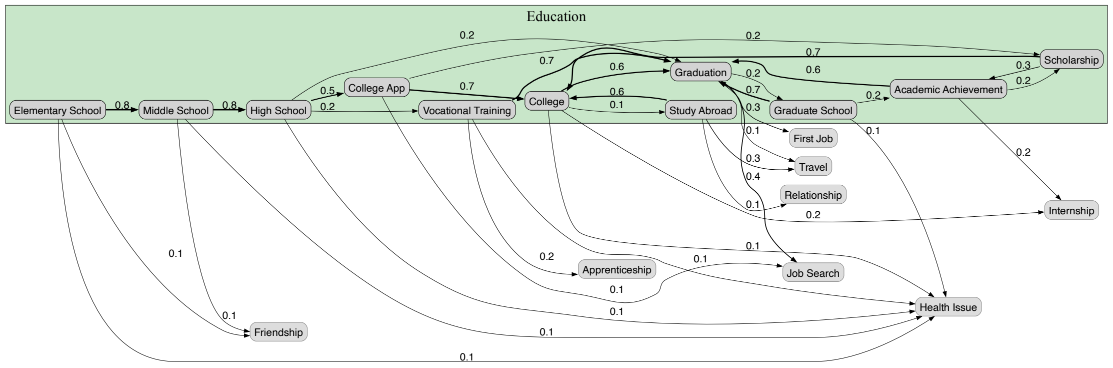

Figure 6: The Education life stage, detailing the probabilistic paths from elementary school through higher education. Nodes within the colored block represent educational milestones, while gray nodes indicate transitions to external stages like Career or Health. The numbers and thickness of the arrows correspond to the transition probabilities, highlighting the primary path towards graduation while showing alternative routes like vocational training or internships.

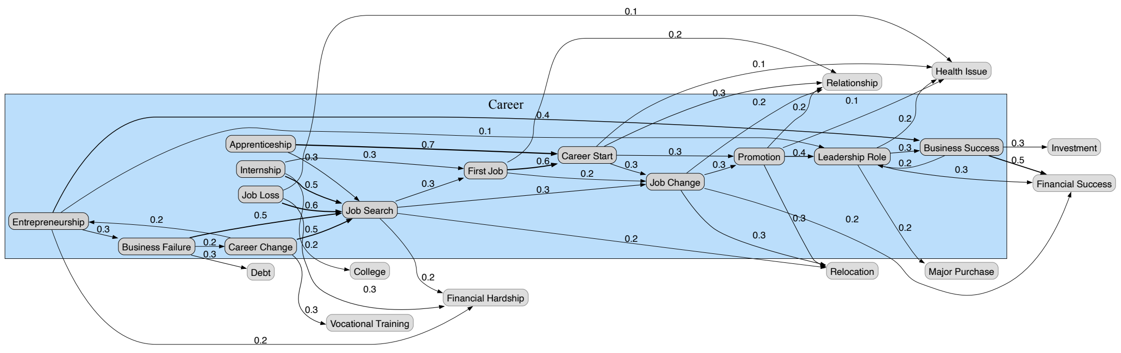

Figure 7: A model of the Career life stage, illustrating the dynamic and non-linear nature of professional life. The diagram shows the typical progression from a first job to a career start, but also includes significant feedback loops such as job loss and job searching. The probabilities on the edges guide the likelihood of events like promotions, career changes, or entrepreneurial ventures, with gray nodes showing strong connections to financial and relationship outcomes.

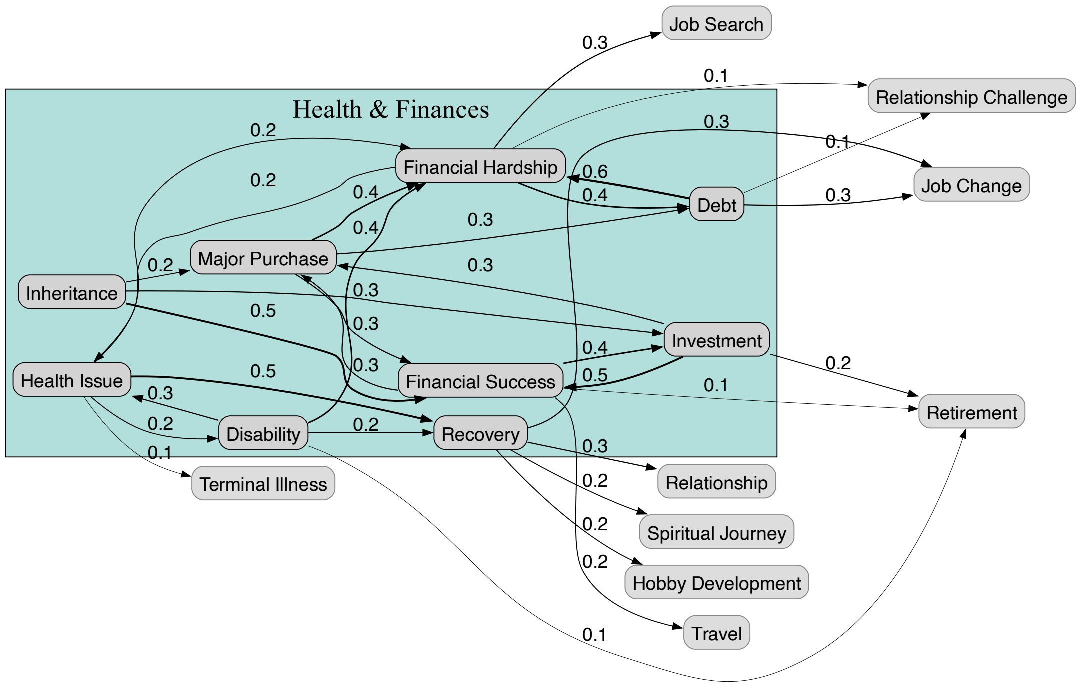

Figure 8: The Health & Finances block, a central hub that influences all other aspects of life. This model visualizes the critical interplay between health events and financial stability. It contains feedback loops for both positive outcomes, like investment leading to financial success, and negative challenges, such as a health issue leading to financial hardship. The gray nodes show how these events directly impact career, relationships, and later life stages.

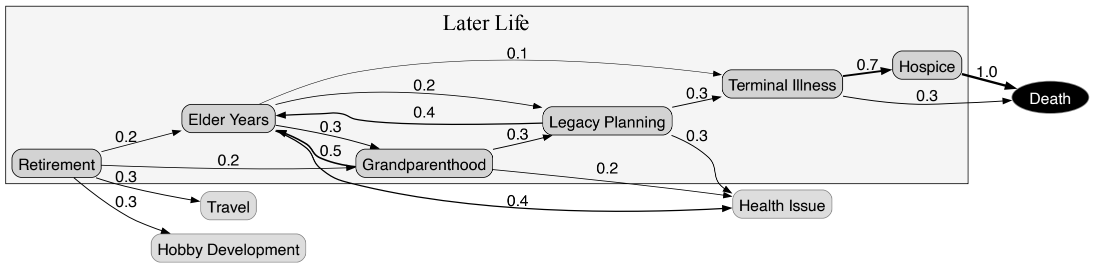

Figure 9: The Later Life stage, modeling the period from retirement to the end of life. The diagram illustrates the shift in focus post-career towards travel, hobbies, and grandparenthood. It also shows the increasing probability of health issues, which create a clear, probabilistic funnel through terminal illness and hospice care to the model's terminal state, Death.

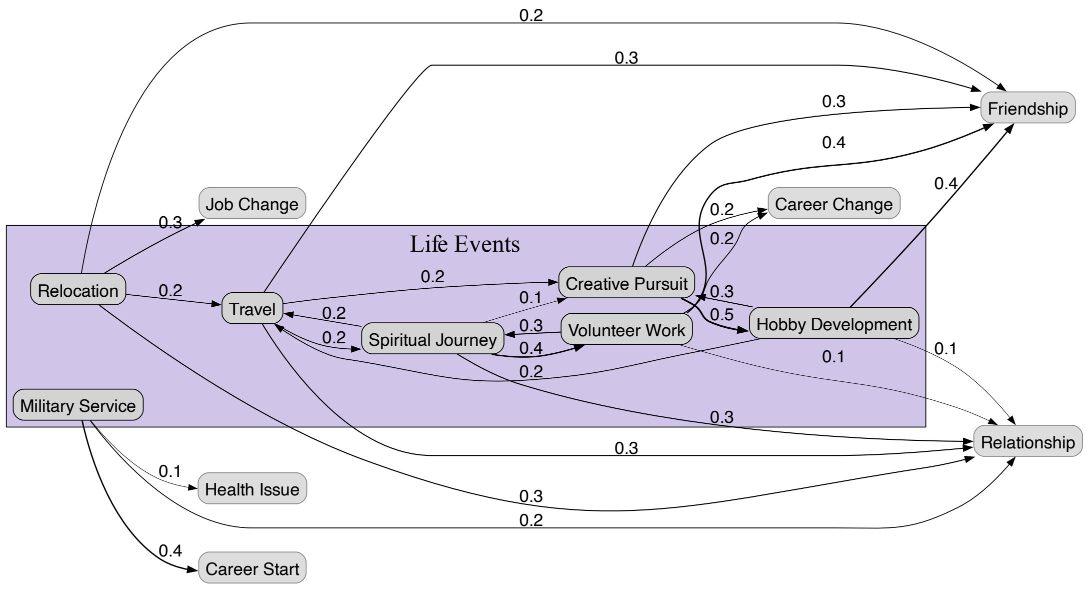

Figure 10: The Life Events block, representing significant personal milestones and pursuits that shape the life path. These events, such as relocation, travel, or military service, often act as catalysts that connect or alter the course of an individual's career and relationships. The graph shows how these pursuits are interconnected and lead to personal development or changes in other life stages.

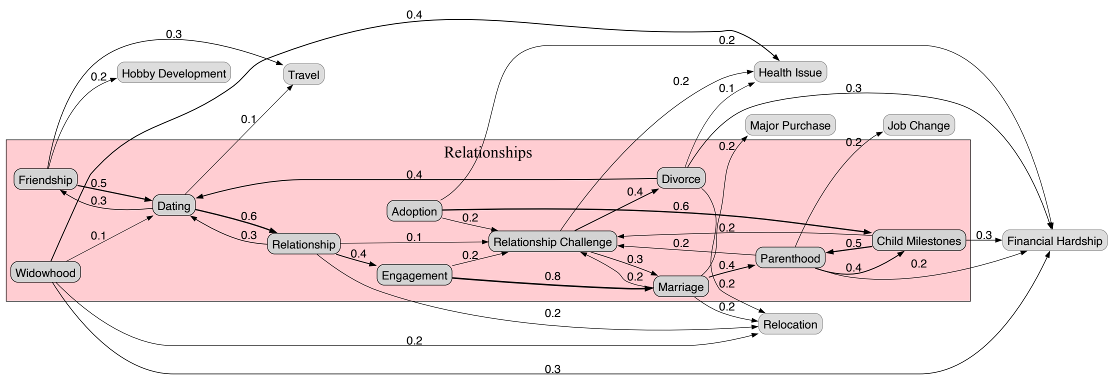

Figure 11: The Relationships life stage, mapping the progression of romantic and family connections. The model follows a common path from dating to marriage and parenthood, with probabilities indicating the likelihood of each step. Crucially, it also incorporates potential setbacks such as relationship challenges or divorce, which create cycles that can return an individual to earlier stages.

Quantitative Evaluation of Additivity

We quantitatively validate our linear additive hypothesis by measuring the deviation from ideal additivity as a function of sequence length, as shown in Figure 12. In this test, random sequences of increasing length were generated and we compute the cosine similarity between the final hidden state produced by the model's recurrent updates and the ideal state calculated via a direct vector sum of the event embeddings. The results from Figure 12 confirm a strong additive structure, with the observed gradual decay in similarity for longer sequences being an expected trade-off for the numerical stability provided by the norm clipping regularization discussed in Section 6.

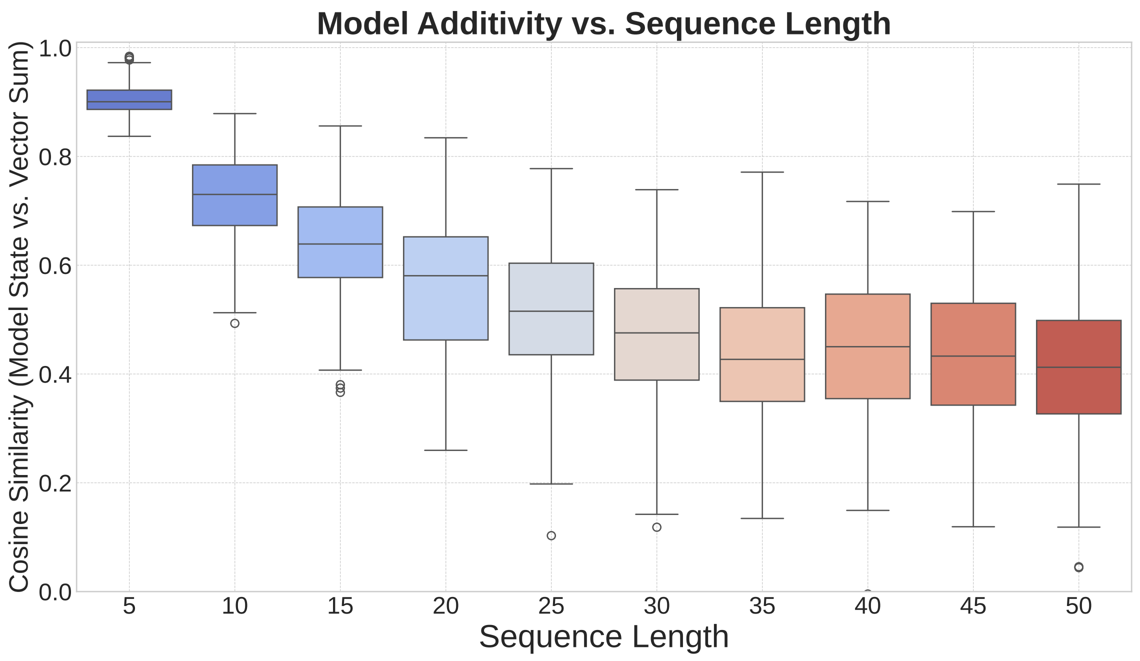

Figure 12: This plot validates our model's compositionality. Cosine similarity between the final hidden state and the ideal vector sum of its parts remains high (0.7), even for long sequences.

Detailed Justification of Analogical Embedding

The analogical reasoning results, presented in the Table 1, show the ability to capture data-driven relationships that go beyond simple chronological order. By analyzing the **Analogy Query** for each **ID**, we can interpret the geometric logic behind the Model's Result.

In **1A**, the query attempts to map the transition from birth to school onto a career context. However, the model interprets the vector 'elementary_school' - 'birth' as the literal concept of "childhood development". When this vector is illogically added to 'first_job', the model defaults to the most representative childhood event it knows, 'late_childhood', although with a moderate cosine similarity of 0.4504. Conversely, in **1B**, the model correctly interprets the vector ‘death‘ - ‘retirement‘ as a ”final life transition”. Applying this concept to ‘graduation‘, it identifies ‘internship‘ as the corresponding next step, representing the transition from academic life to professional life.

The tests in group 2 explore professional trajectories. In **2A**, the model associates the vector for career advancement ‘promotion‘ - ’career_start‘ with ‘military_service‘. This suggests the model has learned that both a promotion and military service are significant, structured steps that follow initial employment, resulting in a relatively high cosine similarity of 0.5048. Analogy test case **2B** shows the model clustering concepts of major positive financial events where the vector representing the outcome of entrepreneurship, when applied to an investment, logically yields another major financial event ‘inheritance‘ with high similarity score 0.5976.

Finally, the tests in group 3 examine relationship dynamics. Row **3A** shows a good example, where the model correctly identifies ‘adoption‘ as a direct parallel to ‘parenthood‘ when applying the ”formalization” vector from ‘marriage‘ - ‘engagement‘. In row **3B**, the model displays a fascinating insight: when asked to find the outcome of financial hardship by applying the vector for a relationship ending (‘divorce‘ - ‘relationship_challenge‘), it returns ‘major_purchase‘. This indicates the model has likely learned from the data that a major purchase is a common cause of financial hardship, linking the two events causally rather than sequentially.