Table 2

Cluster analysis of final hidden states for various RNN configurations. Silhouette scores for state space partitioning and final states clustering, evaluated in both the original high-dimensional state space and the intrinsic low-dimensional (*id*-projected) space. Statistics for cluster centroid distances ($\bar{d}$, $\sigma(d)$) and cluster radii ($\bar{R}$, $\sigma(R)$). Mean cosine similarity between readout vectors and cluster centroids (readout-cluster alignment).

RNN	Embedding	Hidden	Space Partition (Silh.)		Final Clusters (Silh.)		Cluster Distances		Cluster Radii		Readout-Cluster
cell type	dimension	dimension	Original	*id*-projected	Original	*id*-projected	$\bar{d}$	$\sigma(d)$	$\bar{R}$	$\sigma(R)$	alignment
GRU	10	12	0.52	0.53	0.74	0.78	4.04	0.50	0.64	0.20	0.95
GRU	16	16	0.51	0.52	0.77	0.79	4.75	0.56	0.75	0.21	0.95
GRU	16	20	0.51	0.54	0.77	0.79	5.33	0.72	0.82	0.24	0.94
LSTM	10	12	0.53	0.54	0.80	0.81	4.35	0.45	0.64	0.26	0.93
LSTM	16	16	0.52	0.54	0.81	0.83	5.08	0.54	0.69	0.26	0.96
LSTM	16	20	0.51	0.52	0.79	0.82	5.72	0.60	0.86	0.33	0.95
Vanilla	10	12	0.49	0.51	0.72	0.74	4.14	0.48	0.79	0.25	0.94
Vanilla	16	16	0.50	0.52	0.72	0.75	4.83	0.56	0.91	0.29	0.94
Vanilla	16	20	0.49	0.52	0.72	0.76	5.46	0.49	1.05	0.31	0.95

and the *input Jacobian* $\mathbf{J}^{inp}\mathbf{F}$ quantifies the sensitivity of the system to input tokens. The elements of these matrices are defined as:

$$J^{rec}_{ij}\mathbf{F} = \left.\frac{\partial F_i}{\partial h_j}\right|_{(\mathbf{h}_e,\mathbf{x}_e)} \qquad J^{inp}_{ij}\mathbf{F} = \left.\frac{\partial F_i}{\partial x_j}\right|_{(\mathbf{h}_e,\mathbf{x}_e)} \tag{6}$$

When the expansion point corresponds to a fixed point $\mathbf{h}^*$, such that $\mathbf{h}^* = \mathbf{F}(\mathbf{h}^*, \mathbf{x})$, Equation 5 reduces to a linear dynamical system:

$$\Delta\mathbf{h}_t = \mathbf{h}_t - \mathbf{h}_e \approx \mathbf{J}^{rec}\mathbf{F}|_{(\mathbf{h}^*,\mathbf{x}^*)}\Delta\mathbf{h}_{t-1} + \mathbf{J}^{inp}\mathbf{F}|_{(\mathbf{h}^*,\mathbf{x}^*)}\Delta\mathbf{x}_t \tag{7}$$

Due to time-dependent inputs $\mathbf{x_t}$, RNNs are inherently non-autonomous dynamical systems. This nonautonomy requires advanced mathematical tools for analysis. A reverse engineering approach simplifies the analysis by splitting it into three steps: a) identify the topological structure of fixed points under constant input $\mathbf{x}_t$, b) analyze the linearized system dynamics around these fixed points, and c) examine how nonconstant inputs influence and alter (a.k.a. deflect) the system behavior. [3].

In this section, we focus on the first step: identifying the fixed point structure. Fixed points $\mathbf{h}^*_i$ are defined as states satisfying $\mathbf{h}^*_i = \mathbf{F}(\mathbf{h}^*_i, \mathbf{x})$ where $\mathbf{x}$ is a constant input. Typically, $\mathbf{x}$ is set to $\mathbf{0}$, representing the system's autonomous evolution without external inputs. In related work, very slow motion points [55] are considered as approximate fixed points. Here, we adopt this broader definition and define fixed points such that $\mathbf{h}^*_i \approx \mathbf{F}(\mathbf{h}^*_i, \mathbf{0})$. Different procedures have been proposed to identify fixed points [28, 55]. To numerically identify these points, we minimize the speed q of a point in the state space, defined as the squared magnitude of the displacement generated by $\mathbf{F}$.

$$q = \frac{1}{2}\|\mathbf{h} - \mathbf{F}(\mathbf{h}, \mathbf{0})\|_2^2 \tag{8}$$

A numerical optimization process identifies slow-motion ($q < 10^{-8}$) and zero-motion points [14]. To account for different regions of the state space, this procedure is run with multiple initial conditions. In our experiments, over 25K initial states were extracted from trajectories of the SNIPS test dataset. Under the assumption of $\mathbf{x} = \mathbf{0}$, Equation 7 simplifies to:

$$\Delta\mathbf{h}_t = \mathbf{h}_t - \mathbf{h}_e \approx \mathbf{J}^{rec}\mathbf{F}|_{(\mathbf{h}^*,\mathbf{0})}\Delta\mathbf{h}_{t-1} \tag{9}$$

The stability of each fixed point $\mathbf{h}^*_\mathbf{i}$ is determined by the eigenvalues of $\mathbf{J}^{rec}\mathbf{F}|_{(\mathbf{h}^*_i,\mathbf{0})}$. Table 3 presents the fixed points identified for various RNN configurations trained on the 7-class SNIPS dataset, both stable and saddle fixed points

are identified. The number of critical points depends on the type of recurrent cell as well as the dimensions of the embedding and hidden layers. The presence of saddle points with indices higher than one indicates a hierarchy of critical points that trajectories traverse during sentence processing. We further analyze the spatial arrangement of these fixed points by projecting them onto the *id*-top principal components of the state space. The distances δ_s, δ_1, and δ_2 (corresponding to stable points, 1-index saddles and 2-index saddles, respectively) were calculated from each fixed point to the origin. Table 3 summarizes the mean $\bar{\delta}$ and standard deviation $\sigma(\delta)$ of these distances for different network configurations. The distances vary with the embedding and hidden layer dimensions, reflecting how architectural parameters influence the state space geometry.

Table 3

Summary of approximated fixed points ($q < 10^{-8}$) identified in RNNs trained on the SNIPS dataset. The table list the number of attractors, 1-index saddle points, and higher-index saddle points for different network configurations. The mean ($\bar{\delta}$) and standard deviation ($\sigma(\delta)$) of distances from the origin to attractors (δ_s), 1-index saddle points (δ_1) and 2-index saddles (δ_2) are presented for each configuration.

RNN	Embedding	Hidden	Stable	Saddle points		Stable points		1-index		2-index	
cell type	dimension	dimension	points	1-index	Higher-index	$\bar{\delta}_s$	$\sigma(\delta_s)$	$\bar{\delta}_1$	$\sigma(\delta_1)$	$\bar{\delta}_2$	$\sigma(\delta_2)$
Vanilla	10	10	5	9	7	3.93	0.10	3.52	0.13	3.07	0.31
Vanilla	16	16	4	5	4	4.73	0.29	4.25	0.38	4.03	0.36
GRU	10	10	5	6	1	3.72	0.32	3.43	0.33	1.72	-
GRU	16	16	3	3	1	4.71	0.15	4.27	0.18	3.29	-

7.6. Generalizability and Explanatory Power: A Case Study on the ATIS Dataset

To address the generalizability of findings beyond the balanced SNIPS dataset, we replicated our analysis on the ATIS dataset. As detailed in section 5, ATIS presents a more realistic and challenging scenario characterized by a severe class imbalance, with 26 intents (vs. 7 in SNIPS) and the top intent (*flight*) accounting for 73,7% of samples. This allows us to test if our dynamical systems framework can provide insights under less-controlled conditions.

We trained a GRU(emb:64,hid:64) architecture on the imbalanced ATIS dataset. To clarify the scope of the analysis: the full ATIS dataset contains 26 intents, but the standard training set contains only 22 of these. The 4 intents that appear only in the test dataset were excluded from our analysis as they represent an out-of-scope task. Furthermore, 6 of the 22 training intents have no samples in the provided test dataset. Our analysis, therefore, focuses on the 16 intents (shown in Table 4) for which samples were present in both the training and the test data. On this 16-class task, the model achieved a high aggregate accuracy (93.7%), this single metric obscures severe performance disparities. For instance, as shown in Table 4, high-frequency intents like *flight* achieve an F1-score of 0.97, while low-frequency intents like *meal* (6 training samples) fail completely with a 0.00 F1-score. Standard metrics show that the model fails on rare classes, but our framework can help explain why by analyzing the learned state space.

Consistent with our findings on SNIPS, the computation of a GRU(emb:64,hid:64) on ATIS operates on a low-dimensional manifold. A PCA analysis of the test hidden states revealed that only 9 components are required to explain 95% of the variance, a dimensionality far lower than the model's actual hidden and embedding dimension (64) and the number of classes (16). However, the geometry of this manifold is significantly influenced by the class imbalance. As visualized in the 2D and 3D projections in Figure 8 (a) and (b), the state space is dominated by large, well-separated clusters for high-frequency intents (e.g. *flight*, *airfare*). In contrast, most low-frequency intents do not form distinct clusters but instead coalesce into a dense, undifferentiated region.

This "geometric collapse" of rare classes makes standard clustering algorithms like K-means ill-suited for this analysis. K-means, which seeks to find well-defined centers, would fail to partition this dense region and would misrepresent the landscape. Therefore, to properly quantify the quality of the partition the network was supposed to learn, we used a methodology based on the ground-truth labels. We posit that this geometric structure provides a powerful lens to diagnosing classification performance. We use this lens to test our hypothesis that successful classification requires the network to accomplish two distinct tasks, which we evaluated consistently within the 95% variance projected space:

- **Geometric Separation:** The network must guide trajectories for a given intent to a unique and coherent region of the state space. This dynamic steering process is visualized in Figure 8 (c), which contrasts a successful *flight* trajectory with a failed one for a rare intent.
- **Readout Alignment:** The network must correctly align the corresponding readout vector (r_i) with that specific geometric region (c_i). This static link between the final region and the output layer is visualized for the high-frequency intents in Figure 8 (d).

Table 4

Per-intent performance and state space diagnostics for a GRU(emb:64,hid:64) model trained on the ATIS dataset. F1-score and the two metrics from our framework are shown: Silhouette Score (Geometric Separation) and Cosine Similarity (Readout Alignment), computed within the 95% PCA projected state space.

Intent Class	**Test #**	**Train #**	**Performance**			**Projected state space (@95%)**	
			Precision	**Recall**	**F1-score**	**Silh. Score**	**Cosine Sim.**
flight	632	3666	0.96	0.99	0.97	0.57	0.95
airfare	48	423	0.94	0.98	0.96	0.61	0.89
ground_service	36	255	0.84	1.00	0.91	0.76	0.88
airline	38	157	0.90	0.92	0.91	0.62	0.92
abbreviation	33	147	0.97	1.00	0.99	0.50	0.95
flight_time	1	54	1.00	1.00	1.00	0.00	0.90
airport	18	20	0.93	0.78	0.85	0.14	0.94
capacity	21	16	1.00	0.76	0.86	0.31	0.80
aircraft	9	81	0.75	0.67	0.71	-0.13	0.91
ground_fare	7	18	0.50	0.57	0.53	-0.07	0.80
quantity	3	51	0.38	1.00	0.55	0.77	0.88
flight_no	8	12	1.00	0.25	0.40	0.56	0.57
flight+airfare	12	21	0.83	0.42	0.56	0.03	0.66
city	6	19	0.67	0.33	0.44	0.02	0.24
distance	10	20	1.00	0.20	0.33	-0.20	0.42
meal	6	6	0.00	0.00	0.00	-0.21	0.38

A failure in either step can lead to a poor F1-score. Analyzing these two metrics, as shown in Table 4, reveals distinct, interpretable patterns of model behavior that are invisible to a standard F1-score analysis. These four patterns are visualized in Figure 9:

- **Pattern 1: Convergent High Performance** (High F1, High Separation, High Alignment). These intents (e.g. *flight*, *airfare*, *airline*, *ground_service*, *abbreviation*) represent robust learning. They have high F1-scores, a direct consequence of success in both steps. This ideal case is visualized in Figure 9 (a), showing distinct clusters (with high Silhouette scores) and strongly aligned readout vectors.
- **Pattern 2: Geometric Collapse** (Low F1, Low Separation, Low Alignment). These intents (e.g. *meal*, *distance*, *city*, *flight+airfare*) represent a total mechanistic failure. Their near-zero F1-scores are correlated with a failure in both metrics. As shown in Figure 9 (b), these intents fail to form a coherent cluster (negative or near-zero Silhouette scores) and collapse into a dense, mixed region. This is accompanied by a failure in the second step, as shown by their very low readout alignment scores.
- **Pattern 3: Alignment Failure** (Low F1, High Separation, Low Alignment). This pattern is clearly exemplified by *flight_no*. Figure 9 (c) illustrates this pattern perfectly: an intent forms a reasonably distinct cluster (Silhouette 0.56), yet has a very low F1-score (0.40). Table 4 and the visualization both identify the cause: a poor readout alignment (0.57). The network successfully grouped this intent, but failed to map the correct output neuron to that group.
- **Pattern 4: Alignment-Driven Classification** (High F1, Low Separation, High Alignment). This is the inverse finding, seen in intents like *aircraft*, *airport*, *capacity* and *flight_time*). These intents achieve high F1-scores